simesh Matlab toolbox, User's Guide*

version 0.4.7

François Cuvelier†

January 30, 2025

Abstract

This object-oriented Matlab toolbox allows to use simplial meshes generated from `gmsh` (in dimension 2 or 3) or an hypercube triangulation (in any dimension). For graphical representation the siplt toolbox is used.

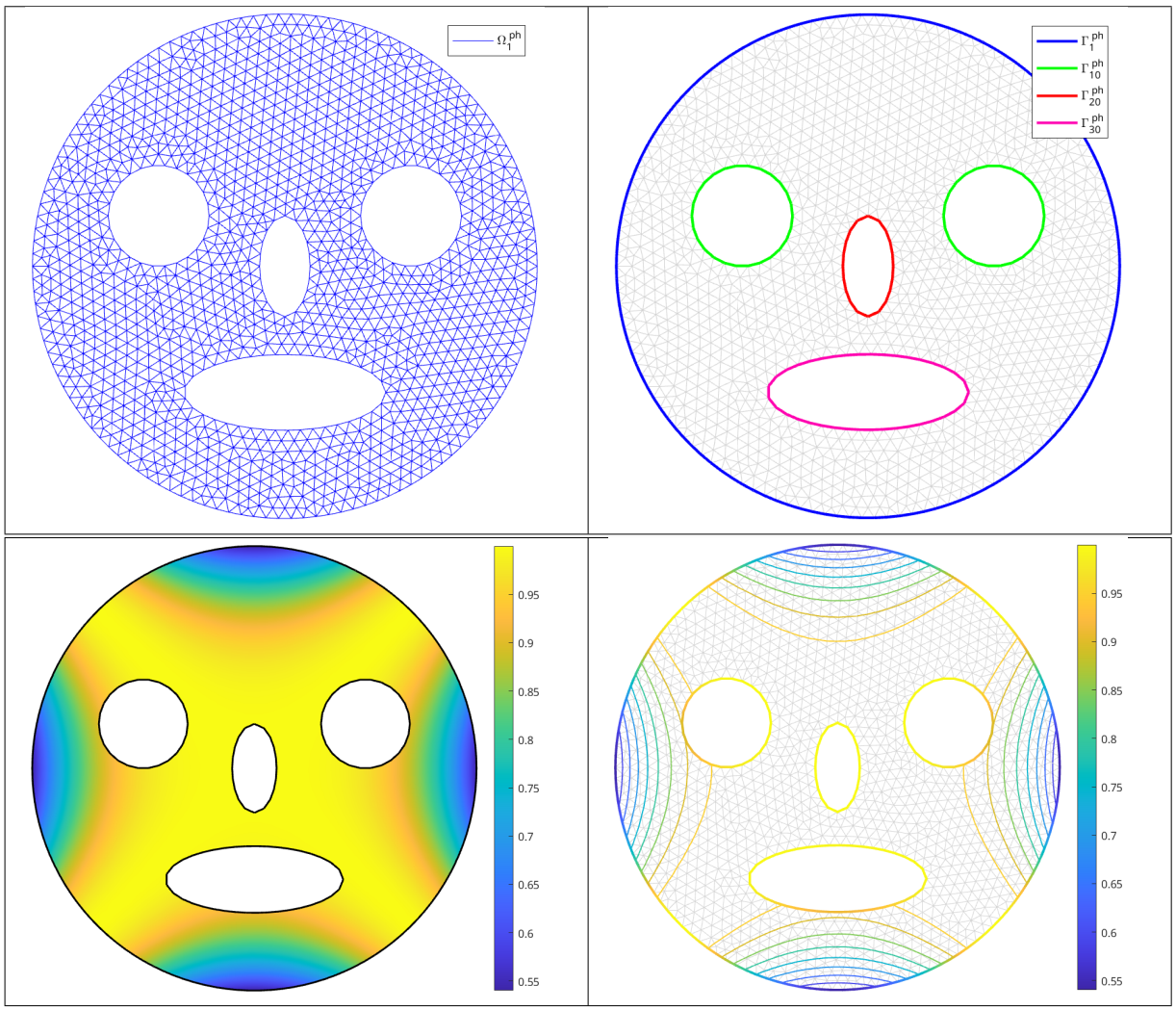
0 Contents

*LaTeX manual, revision 0.4.7, compiled with Matlab 2022a, and toolboxes `fc-simesh`[0.4.7], `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-siplt`[0.2.6]

†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.

1 Introduction

The fc simesh Matlab toolbox was created to simplify the use of simplicial meshes and to easily represent data on all or parts of them. In 2D or 3D `gmsh` can be used under Matlab to build triangular or tetrahedral meshes by using the fc oogmsh toolbox[1]. Thereafter the mesh stored as a file (.msh) can be read by using the siMesh object. In Listing 1, a 2D example is provided with the 4 generated figures. For graphic representations, the fc siplt toolbox[2] is used by the siMesh object Th as follows Th.**plotmesh**(...) is **fc_siplt.plotmesh**(Th,...), Th.**plot**(...) is **fc_siplt.plot**(Th,...) and so on.




```
close all
geofile=fc_simesh.get_geo(2,2,'sample2D01.geo');
% Using GMSH >= 4.0.0 to create mesh file
meshfile=fc_oogmsh.gmsh.buildmesh2d(geofile,200,'force',true);
% Creating siMesh object by reading the mesh file
Th=fc_simesh.siMesh(meshfile);
% Computing datas on siMesh object
u=@(x,y) cos(x.^2-y.^2);
U=Th.eval(u);
% Graphics
figure(1)
Th.plotmesh('inlegend',true)
axis image;axis off
fc_graphics4mesh.legend()

figure(2)
Th.plotmesh('color','LightGray')
hold on
Th.plotmesh('d',1,'inlegend',true,'LineWidth',2)
axis image;axis off
fc_graphics4mesh.legend()

figure(3)
Th.plot(U,'plane',true)
colorbar
shading interp
axis image;axis off
hold on
Th.plotmesh('d',1,'LineWidth',1.5,'color','k')

figure(4)
Th.plotmesh('color','LightGray')
axis image;axis off
hold on
Th.plot(U,'d',1,'LineWidth',2,'plane',true)
colorbar
Th.plotiso(U,'niso',10,'LineWidth',1,'plane',true)
```

Listing 1: **fc_simesh**.demos.sample2D01 script with figure 1 (top left), figure 2 (top right), figure 3 (bottom left) and figure 4 (bottom right).

In higher dimension the hypermesh toolbox[3] can be used to obtain meshes of an hypercube by using the **fc_simesh**.constructor.hypercube function.

2 Installation

This toolbox was only tested on Ubuntu 24.04.1 with Matlab R2022a.

One just has to get/download the install file

mfc_simesh_install.m

or get it on the dedicated web page. Thereafter, one run it under Matlab. This command download, extract and configure the simesh toolbox and all the required required toolboxes in the current directory.

For example, to install this toolbox in `~/Matlab` directory, one has to copy the file mfc_simesh_install.m in the `~/Matlab` directory. Then in a Matlab terminal run the following commands

```
>> cd ~/Matlab
>> mfc_simesh_install
```

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

```
Parts of the <fc-simesh> Matlab toolbox.
Copyright (C) 2017-2025 F. Cuvelier

1- Downloading and extracting the toolboxes
2- Setting the <fc-simesh> toolbox
Write in ~/Matlab/fc-simesh-full/fc_simesh-0.4.7/configure_loc.m ...
3- Using toolboxes :
   ->             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-siplt : 0.2.7
 with            fc-simesh : 0.4.7
*** Using instructions
   To use the <fc-simesh> toolbox:
   addpath('~/Matlab/fc-simesh-full/fc_simesh-0.4.7')
   fc_simesh.init()

   See ~/Matlab/mfc_simesh_set.m
```

The complete toolbox (i.e. with all the other needed toolboxes) is stored in the directory `~/Matlab/fc-simesh-full` and, for each Matlab session, one have to set the toolbox by:

```
>> addpath('~/Matlab/fc-simesh-full/fc-simesh-0.4.7')
>> fc_simesh.init()
```

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

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

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

```
Using fc_simesh[0.4.7] 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_siplt[0.2.7].
[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

```
~/Matlab/fc-simesh-full
```

3 Mesh Objects

In geometry, a simplex is a generalization of the notion of a triangle or tetrahedron to arbitrary dimensions. Specifically, a k-simplex in $\mathbb{R}^{\mathrm{dim}}$, $0 \leqslant k \leqslant \mathrm{dim}$, is a polytope which is the convex hull of its $k+1$ vertices of $\mathbb{R}^{\mathrm{dim}}$. More formally, suppose the $k+1$ vertices $\mathrm{q}^0, \cdots, q^k \in \mathbb{R}^{\mathrm{dim}}$ such that $\mathrm{q}^1 - \mathrm{q}^0, \cdots, \mathrm{q}^k - \mathrm{q}^0$ are linearly independent. Then, the k-simplex K determined by them is the set of points

$$K = \left\{ \sum_{i=0}^{k} \lambda_i \mathrm{q}^i \mid \lambda_i \geqslant 0,\ i \in [\![0, k]\!],\ \text{with} \sum_{i=0}^{k} \lambda_i = 1 \right\}.$$

We denote by **k-simplicial elementary mesh** in $\mathbb{R}^{\mathrm{dim}}$, $0 \leqslant k \leqslant \mathrm{dim}$, a mesh with **unique geometrical label** only composed with k-simplices.

A **d-simplicial mesh** in $\mathbb{R}^{\mathrm{dim}}$, $0 \leqslant \mathrm{d} \leqslant \mathrm{dim}$, is an union of k-simplicial elementary meshes with $k \in [\![0, \mathrm{d}]\!]$.

3.1 fc_simesh.siMeshElt object

An elementary d-simplicial mesh in dimension dim is represented by the class siMeshElt . We give properties of this class :

Properties of siMeshElt object for d-simplicial elementary meshes in $\mathbb{R}^{dim}$

dim	:	integer space dimension
d	:	integer ($0 \leqslant d \leqslant dim$)
nq	:	integer number of vertices
nme	:	integer number of elements (d-simplices)
q	:	dim-by-nq array of reals array of vertex coordinates
me	:	(d + 1)-by-nme array of integers connectivity array for **mesh elements**
vols	:	1-by-nme array of reals array of mesh element volumes
geolab	:	integer geometrical label
label	:	integer same as geolab
physlab	:	integer physical label
h	:	double mesh step size (=maximum edge length in the mesh)
toGlobal	:	1-by-nq array of integers convert from local to global mesh vertices numbering. Prefer the use of toParents{**end**} instead. *It will be removed in a future release.*
toParent	:	1-by-nq array of integers convert from local to parent mesh vertices numbering (same as toGlobal if not part of a partitioned mesh). Prefer the use of toParents{1} instead. *It will be removed in a future release.*
nqParents	:	1-by-*n* array of integers nqParents(1) number of vertices in the *parent* mesh, nqParents(2) number of vertices in the *parent* of the *parent* mesh, nqParents(**end**) number of vertices in the global mesh.
toParents	:	1-by-*n* cell array toParents{1} indices array which convert local vertices numbering to the *parent* mesh vertices numbering, toParents{2} indices array which convert local vertices numbering to the *parent* of the *parent* mesh, toParents{**end**} indices array which convert local vertices numbering to the global mesh.

More precisely

- q(i,j) is the i-th coordinate of the j-th vertex, i $\in \{1,\ldots,dim\}$, j $\in \{1,\ldots,nq\}$. The j-th vertex will be also denoted by $q^j = q(:,j)$.
- me(r,k) is the storage index of the r-th vertex of the k-th element (d-simplex), in the array q, for r $\in \{1,...,d+1\}$ and k $\in \{1,\ldots,nme\}$. So q(:,me(r,k)) represents the coordinates of the r-th vertex of the k-th mesh element.
- vols(k) is the volume of the k-th d-simplex .

3.2 fc_simesh.siMesh object

A d-simplicial mesh in dimension dim, represented as an siMesh object, is an union of siMeshElt objects which are elementary l-simplicial meshes ($0 \leqslant l \leqslant d$) in space dimension dim.

siMesh object properties

dim	:	integer space dimension
d	:	integer d-dimensional simplicial mesh
`sTh`	:	array of siMeshElt objects
`nsTh`	:	number of siMeshElt objects
`sThsimp`	:	array of `nsTh` integers i-th siMeshElt object in `sTh` is a `sThsimp`(i)-simplicial elementary mesh
sThgeolab	:	array of `nsTh` integers in `sTh` geometrical label of i-th siMeshElt object in `sTh` is number sThgeolab(i)
`sThlab`	:	array of `nsTh` integers in `sTh` same as sThgeolab.
sThphyslab	:	array of `nsTh` integers in `sTh` physical label of i-th siMeshElt object in `sTh` is number sThphyslab(i). Could be **NaN** value (i.e. not physical).
nq	:	integer number of vertices in the mesh
toGlobal	:	1-by-nq array of integers convert from local to global mesh vertices numbering. Prefer the use of ndtoParents instead. *It will be removed in a future release.*
`toParent`	:	1-by-nq array of integers convert from local to parent mesh vertices numbering (same as toGlobal if not part of a partitioned mesh). Prefer the use of toParents1 instead. *It will be removed in a future release.*
nqParents	:	1-by-n array of integers Only used with partitioned mesh and the fc-psimesh toolbox.
toParents	:	1-by-n cell array Only used with partitioned mesh and the fc-psimesh toolbox.

Let Th be a siMesh object. The global dim-by-(Th.nq) array q of mesh vertices is not explicitly stored in Th, however one can easily build it if necessary:

```
q=zeros(Th.dim,Th.nq);
for i=Th.find(Th.d)
  q(:,Th.sTh{i}.toParents{1})=Th.sTh{i}.q;
end
```

3.3 Mesh samples

3.3.1 2-simplicial mesh in $\mathbb{R}^2$

Listing 2: : 2D siMesh object from sample20.geo

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('sample20',20,'force',true);
Th=fc_simesh.siMesh(meshfile);
fprintf('*** Th: \n')
disp(Th)
fprintf('*** Th.sTh{9}:\n')
disp(Th.sTh{9})
```

Output

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/2d/sample20.geo
[fc-oogmsh] Overwritting mesh file <fc-oogmsh>/meshes/sample20-20.msh
[fc-oogmsh] Starting building mesh <fc-oogmsh>/meshes/sample20-20.msh with gmsh 4.13.1
[fc-oogmsh] Using command : gmsh -2 -setnumber N 20 -string "Mesh.MshFileVersion=4.1;" <fc-oogmsh>/geodir/2d/sample20.geo -o ...
    <fc-oogmsh>/meshes/sample20-20.msh
 Be patient...
[fc-oogmsh] Using gmsh 4.13.1 to write MSH file format version 4.1 in <fc-oogmsh>/meshes/sample20-20.msh
*** Th:
  fc_simesh.siMesh with properties:
              d: 2 double
            dim: 2 double
             sTh: (1x20 cell)
           nsTh: 20 double
       toGlobal: (1x2327 double)
       toParent: (1x2327 double)
        sThsimp: [ 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 ] (1x20 double)
         sThlab: [ 105 110 115 120 101 102 103 104 106 107 108 109 111 112 113 114 116 117 118 119 ] (1x20 double)
      sThcolors: (20x3 double)
           bbox: [ -1 1 -1 1 ] (1x4 double)
      sThgeolab: [ 105 110 115 120 101 102 103 104 106 107 108 109 111 112 113 114 116 117 118 119 ] (1x20 double)
     sThphyslab: [ 10 20 1 2 101 102 103 104 20 20 20 20 1 1 1 1 2 2 2 2 ] (1x20 double)
    sThpartlabs: []
   sThboundlabs: []
             nq: 2327 double
      nqParents: 2327 double
      toParents: (1x1 cell)
          other: (1x1 struct)
*** Th.sTh{9}:
  siMeshElt with properties:
              d: 1 double
            dim: 2 double
             nq: 8 double
            nme: 7 double
              q: (2x8 double)
             me: (2x7 double)
       toGlobal: [ 5 6 173 174 175 176 177 178 ] (1x8 double)
       nqGlobal: 2327 double
       toParent: [ 5 6 173 174 175 176 177 178 ] (1x8 double)
       nqParent: 2327 double
     nqParents: 2327 double
     toParents: (1x1 cell)
    elementTags: [ 161 162 163 164 165 166 167 ] (1x7 double)
         label: 106 double
           Tag: (1x28 char)
         color: [ 1 0.827586 0.0344828 ] (1x3 double)
          vols: [ 0.0447858 0.0447858 0.0447858 0.0447858 0.0447858 0.0447858 0.0447858 ] (1x7 double)
      gradBaCo: (7x2x2 double)
        geolab: 106 double
       physlab: 20 double
        pcolor: [ 1 0 0 ] (1x3 double)
       normals: (2x7 double)
     ptxnormals: (2x8 double)
       partlab: []
          bbox: [ 0 0.2 0 0.2 ] (1x4 double)
             h: 0.0447858 double
         order: 1 double
```

From the output of the Listing 2 or from the Figure 1 the complete physical domain (2-simplex elements) with physical labels is

$$\Omega^{\text{ph}} = \Omega_1^{\text{ph}} \cup \Omega_2^{\text{ph}} \cup \Omega_{10}^{\text{ph}} \cup \Omega_{20}^{\text{ph}}$$

and the physical boundary (1-simplex elements) is

$$\Gamma^{\text{ph}} = \Gamma_1^{\text{ph}} \cup \Gamma_2^{\text{ph}} \cup \Gamma_{20}^{\text{ph}} \cup \Gamma_{101}^{\text{ph}} \cup \Gamma_{102}^{\text{ph}} \cup \Gamma_{103}^{\text{ph}} \cup \Gamma_{104}^{\text{ph}}.$$

Each physical element is made of one or more geometrical elements (i.e. elementary mesh):

$$\Omega_1^{\text{ph}} = \Omega_{115}^{\text{geo}},\ \Omega_2^{\text{ph}} = \Omega_{120}^{\text{geo}}, \Omega_{10}^{\text{ph}} = \Omega_{105}^{\text{geo}} \text{ and } \Omega_{20}^{\text{ph}} = \Omega_{110}^{\text{geo}}$$

and

$$\Gamma_1^{\text{ph}} = \Gamma_{111:114}^{\text{geo}},\ \Gamma_2^{\text{ph}} = \Gamma_{116:119}^{\text{geo}},\ \Gamma_{20}^{\text{ph}} = \Gamma_{106:109}^{\text{geo}}, \Gamma_i^{\text{ph}} = \Gamma_i^{\text{geo}}, \forall i \in [101:104].$$

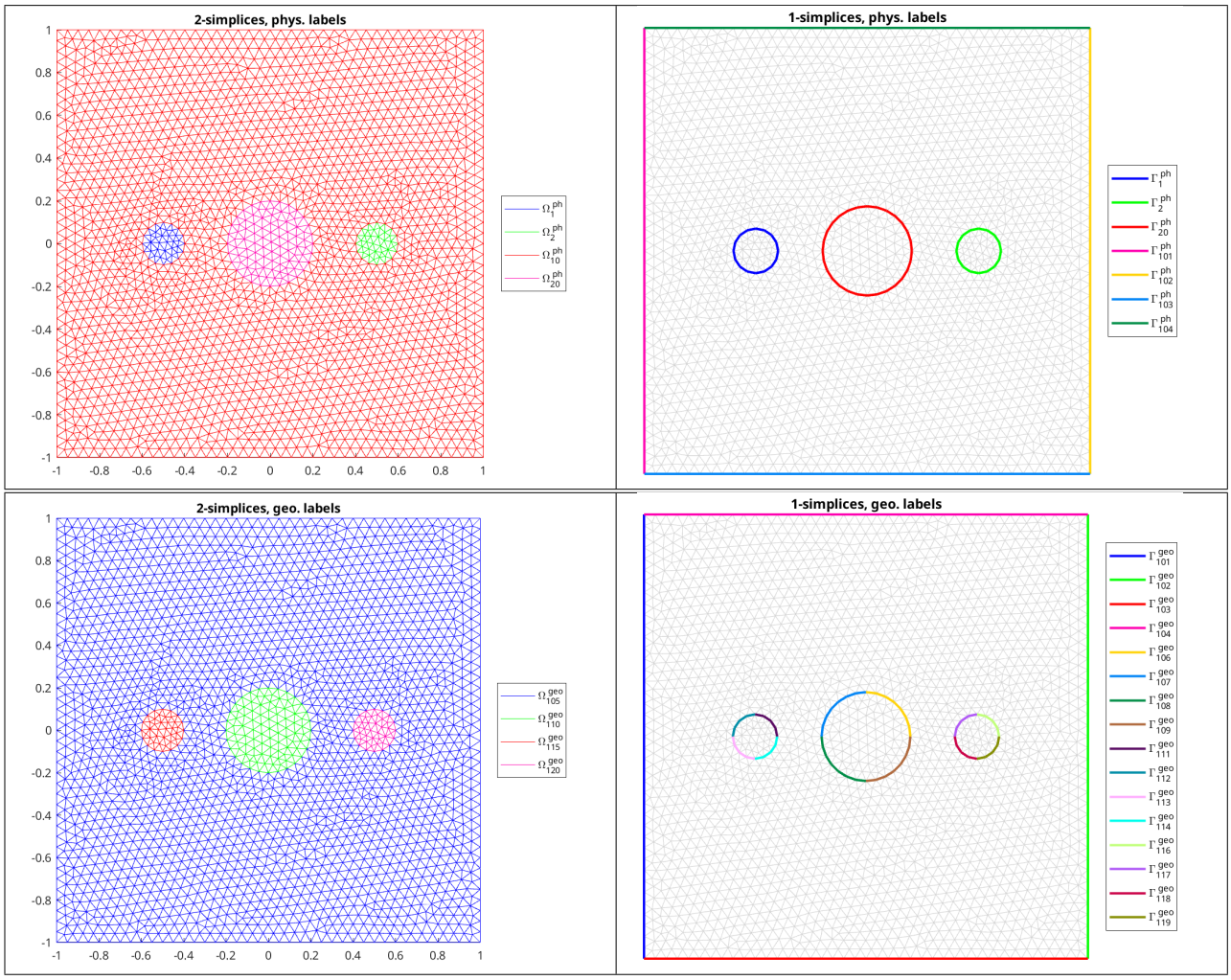


Figure 1: 2D siMesh object from `sample20.geo`, geometrical labels (top) and physical labels (bottom).

3.3.2 Sample of a 3-simplicial mesh in $\mathbb{R}^3$

Listing 3: : 3D Mesh from `quart_sphere2.geo`

```
meshfile=fc_oogmsh.gmsh.buildmesh3d('quart_sphere2',5,'force',true);
Th=fc_simesh.siMesh(meshfile);
fprintf('*** Th: \n')
disp(Th)
fprintf('*** Th.sTh{9}:\n')
disp(Th.sTh{9})
```

Output

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/3d/quart_sphere2.geo
[fc-oogmsh] Overwritting mesh file <fc-oogmsh>/meshes/quart_sphere2-5.msh
[fc-oogmsh] Starting building mesh <fc-oogmsh>/meshes/quart_sphere2-5.msh with gmsh 4.13.1
[fc-oogmsh] Using command : gmsh -3 -setnumber N 5 -string "Mesh.MshFileVersion=4.1;" <fc-oogmsh>/geodir/3d/quart_sphere2.geo -o ...
    <fc-oogmsh>/meshes/quart_sphere2-5.msh
 Be patient...
[fc-oogmsh] Using gmsh 4.13.1 to write MSH file format version 4.1 in <fc-oogmsh>/meshes/quart_sphere2-5.msh
*** Th:
  fc_simesh.siMesh with properties:
              d: 3 double
            dim: 3 double
             sTh: (1x23 cell)
            nsTh: 23 double
        toGlobal: (1x1172 double)
        toParent: (1x1172 double)
         sThsimp: [ 3 3 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 0 0 0 0 0 ] (1x23 double)
          sThlab: [ 25 27 11 13 15 17 19 21 23 1 2 3 4 5 6 7 8 9 1 2 3 4 5 ] (1x23 double)
       sThcolors: (23x3 double)
            bbox: [ -1 1 0 1 0 1 ] (1x6 double)
       sThgeolab: [ 25 27 11 13 15 17 19 21 23 1 2 3 4 5 6 7 8 9 1 2 3 4 5 ] (1x23 double)
      sThphyslab: [ 1 2 1 2 3 7 6 4 5 1 4 3 2 9 7 8 5 6 1 2 3 4 5 ] (1x23 double)
     sThpartlabs: []
    sThboundlabs: []
             nq: 1172 double
      nqParents: 1172 double
      toParents: (1x1 cell)
          other: (1x1 struct)
*** Th.sTh{9}:
  siMeshElt with properties:
              d: 2 double
            dim: 3 double
             nq: 112 double
            nme: 187 double
              q: (3x112 double)
             me: (3x187 double)
        toGlobal: (1x112 double)
        nqGlobal: 1172 double
        toParent: (1x112 double)
        nqParent: 1172 double
      nqParents: 1172 double
      toParents: (1x1 cell)
    elementTags: (1x187 double)
          label: 23 double
            Tag: (1x31 char)
          color: [ 0 0.551724 0.275862 ] (1x3 double)
           vols: (1x187 double)
      gradBaCo: (187x3x3 double)
        geolab: 23 double
        physlab: 5 double
         pcolor: [ 1 0.827586 0.0344828 ] (1x3 double)
        normals: (3x187 double)
     ptxnormals: (3x112 double)
        partlab: []
           bbox: [ -1 0 0 1 0 0 ] (1x6 double)
              h: 0.13635 double
          order: 1 double
```

The mesh obtained from Listing 3 is a 3-simplicial mesh in $\mathbb{R}^3$ and is composed of :

- two 3-simplicial physical meshes : Ω_i^{ph}, $\forall i \in \{1,2\}$ where $\Omega_1^{\text{ph}} = \Omega_{25}^{\text{geo}}$ and $\Omega_2^{\text{ph}} = \Omega_{27}^{\text{geo}}$.
- seven 2-simplicial physical meshes : Γ_i^{ph} $\forall i \in [\![1,7]\!]$ where $\Gamma_1^{\text{ph}} = \Gamma_{11}^{\text{geo}}$, $\Gamma_2^{\text{ph}} = \Gamma_{13}^{\text{geo}}$, $\Gamma_3^{\text{ph}} = \Gamma_{15}^{\text{geo}}$, $\Gamma_4^{\text{ph}} = \Gamma_{21}^{\text{geo}}$, $\Gamma_5^{\text{ph}} = \Gamma_{23}^{\text{geo}}$, $\Gamma_6^{\text{ph}} = \Gamma_{19}^{\text{geo}}$ and $\Gamma_7^{\text{ph}} = \Gamma_{17}^{\text{geo}}$.
- nine 1-simplicial physical meshes : $\partial\Gamma_i^{\text{ph}}$ $\forall i \in [\![1,9]\!]$ where $\partial\Gamma_1^{\text{ph}} = \partial\Gamma_1^{\text{geo}}$, $\partial\Gamma_2^{\text{ph}} = \partial\Gamma_4^{\text{geo}}$, $\partial\Gamma_3^{\text{ph}} = \partial\Gamma_3^{\text{geo}}$, $\partial\Gamma_4^{\text{ph}} = \partial\Gamma_2^{\text{geo}}$, $\partial\Gamma_5^{\text{ph}} = \partial\Gamma_8^{\text{geo}}$, $\partial\Gamma_6^{\text{ph}} = \partial\Gamma_9^{\text{geo}}$, $\partial\Gamma_7^{\text{ph}} = \partial\Gamma_6^{\text{geo}}$, $\partial\Gamma_8^{\text{ph}} = \partial\Gamma_7^{\text{geo}}$, $\partial\Gamma_9^{\text{ph}} = \partial\Gamma_5^{\text{geo}}$,
- five 0-simplicial physical meshes : $\partial^2\Gamma_i^{\text{ph}}$ $\forall i \in [\![1,5]\!]$ where $\partial^2\Gamma_i^{\text{ph}} = \partial^2\Gamma_i^{\text{geo}}$.

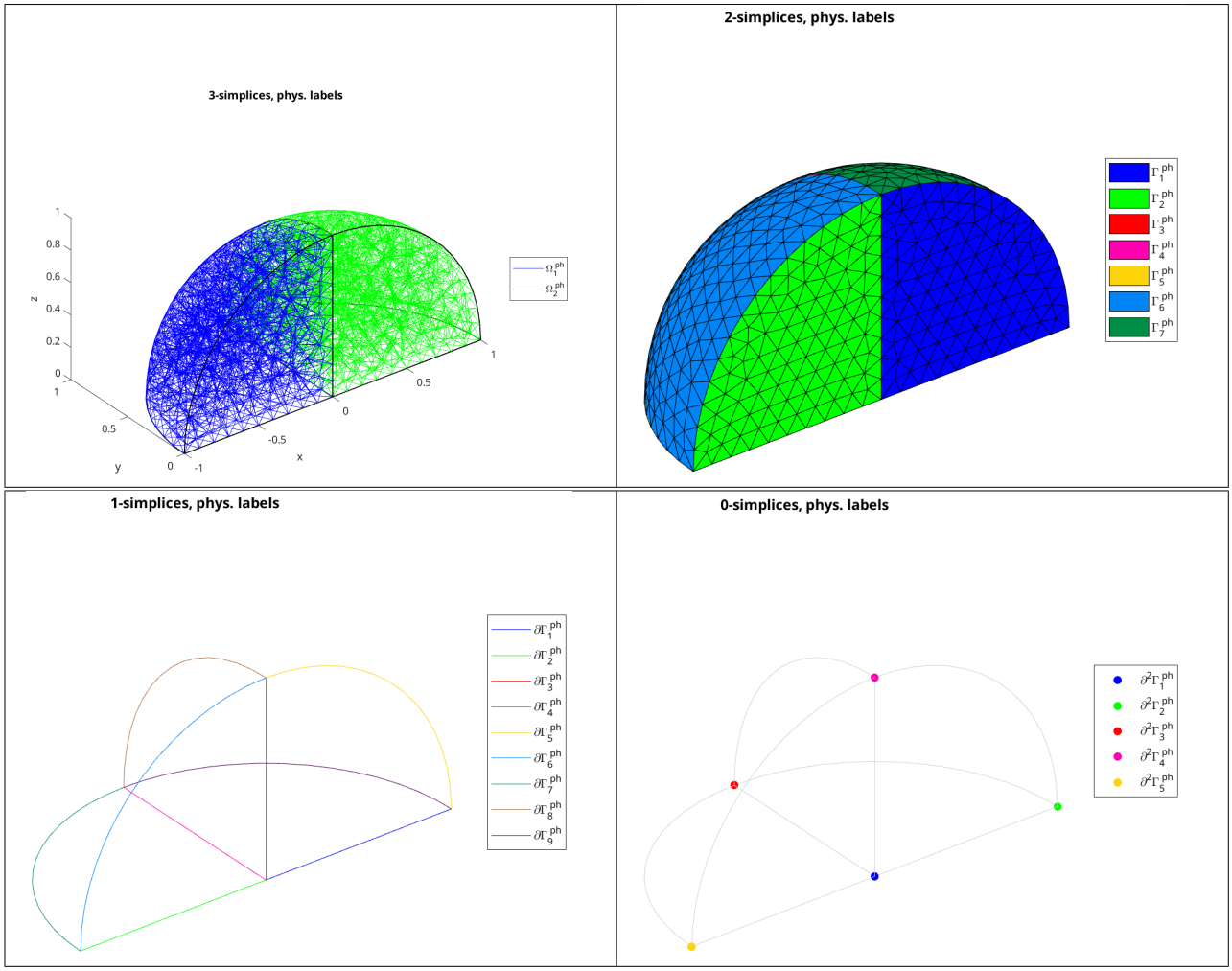


Figure 2: 3D Mesh from `quart_sphere2.geo`, physical labels.

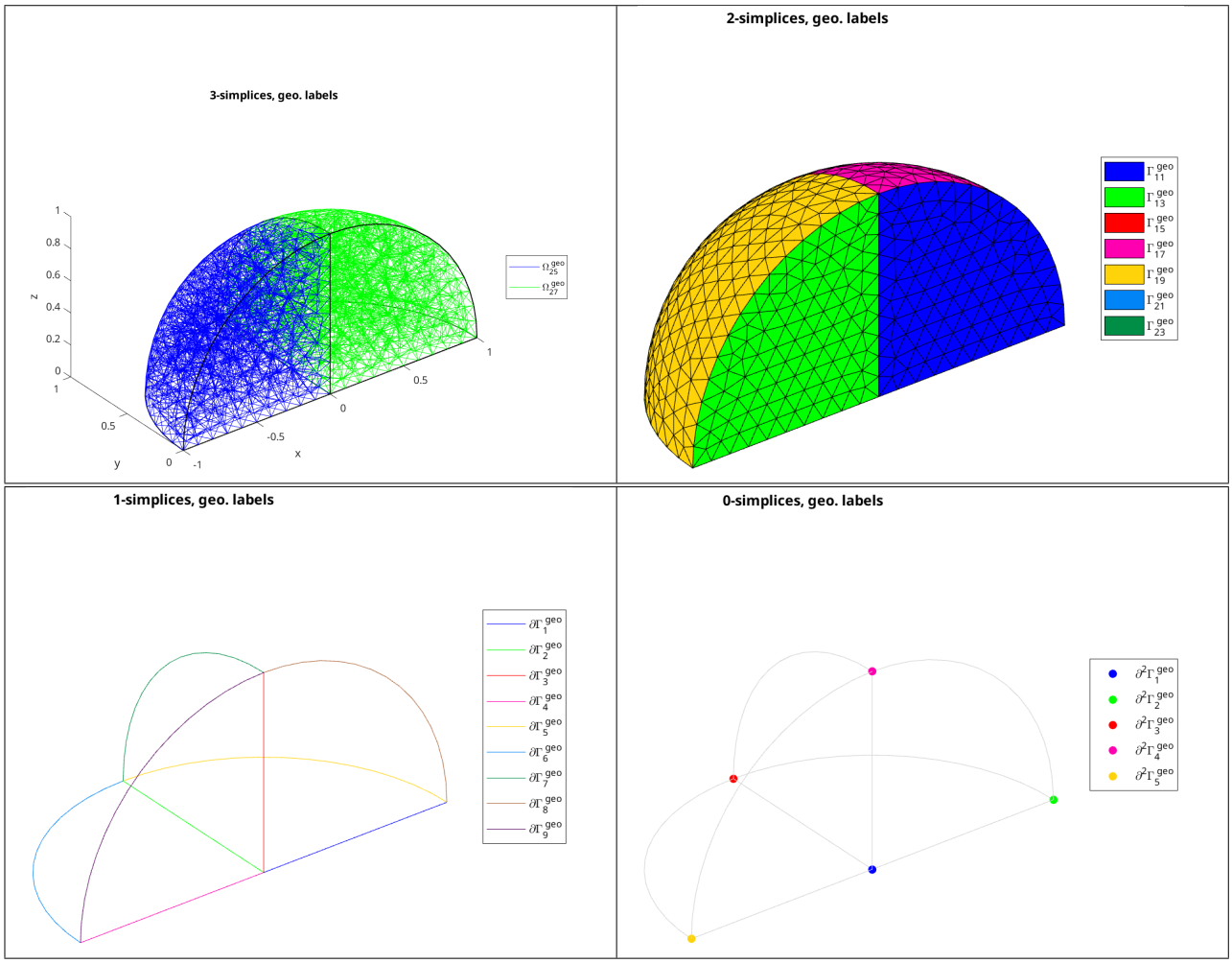


Figure 3: 3D Mesh from `quart_sphere2.geo`, geometrical labels.

3.3.3 Sample of a 2-simplicial mesh in $\mathbb{R}^3$

Listing 4: : 3D surface Mesh from `demisphere4surf.geo`

```
meshfile=fc_oogmsh.gmsh.buildmesh3ds('demisphere4surf',5,'force',true);
Th=fc_simesh.siMesh(meshfile);
fprintf('***_Th:_\n')
disp(Th)
fprintf('***_Th.sTh{9}:\n')
disp(Th.sTh{9})
```

Output

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/3ds/demisphere4surf.geo
[fc-oogmsh] Overwritting mesh file <fc-oogmsh>/meshes/demisphere4surf-5.msh
[fc-oogmsh] Starting building mesh <fc-oogmsh>/meshes/demisphere4surf-5.msh with gmsh 4.13.1
[fc-oogmsh] Using command : gmsh -2 -setnumber N 5 -string "Mesh.MshFileVersion=4.1;" <fc-oogmsh>/geodir/3ds/demisphere4surf.geo -o ...
    <fc-oogmsh>/meshes/demisphere4surf-5.msh
 Be patient...
[fc-oogmsh] Using gmsh 4.13.1 to write MSH file format version 4.1 in <fc-oogmsh>/meshes/demisphere4surf-5.msh
*** Th:
  fc_simesh.siMesh with properties:
             d: 2 double
           dim: 3 double
           sTh: (1x12 cell)
          nsTh: 12 double
      toGlobal: (1x228 double)
      toParent: (1x228 double)
       sThsimp: [ 2 2 2 2 1 1 1 1 1 1 1 1 ] (1x12 double)
        sThlab: [ 1 2 3 4 1 2 3 4 5 6 7 8 ] (1x12 double)
     sThcolors: (12x3 double)
          bbox: [ -1 1 -1 1 0 1 ] (1x6 double)
     sThgeolab: [ 1 2 3 4 1 2 3 4 5 6 7 8 ] (1x12 double)
    sThphyslab: [ 1 2 3 4 1 2 3 4 5 6 7 8 ] (1x12 double)
   sThpartlabs: []
  sThboundlabs: []
            nq: 228 double
     nqParents: 228 double
     toParents: (1x1 cell)
         other: (1x1 struct)
*** Th.sTh{9}:
  siMeshElt with properties:
             d: 1 double
           dim: 3 double
            nq: 9 double
           nme: 8 double
             q: (3x9 double)
            me: (2x8 double)
      toGlobal: (1x9 double)
      nqGlobal: 228 double
      toParent: (1x9 double)
      nqParent: 228 double
    nqParents: 228 double
    toParents: (1x1 cell)
  elementTags: [ 33 34 35 36 37 38 39 40 ] (1x8 double)
        label: 5 double
          Tag: (1x26 char)
        color: [ 1 0.827586 0.0344828 ] (1x3 double)
         vols: [ 0.196034 0.196034 0.196034 0.196034 0.196034 0.196034 0.196034 0.196034 ] (1x8 double)
     gradBaCo: (8x2x3 double)
       geolab: 5 double
      physlab: 5 double
       pcolor: [ 1 0.827586 0.0344828 ] (1x3 double)
      normals: (3x8 double)
   ptxnormals: (3x9 double)
      partlab: []
         bbox: [ -1 0 0 0 0 1 ] (1x6 double)
            h: 0.196034 double
        order: 1 double
```

The mesh obtained from the Listing 4 or from the Figure 4 is a 2-simplicial mesh in $\mathbb{R}^3$ and is composed of :

- four 2-simplicial elementary meshes : $\Omega_i^{\text{ph}} = \Omega_i^{\text{geo}}$, $\forall i \in [\![1, 4]\!]$
- eight 1-simplicial elementary meshes : $\Gamma_i^{\text{ph}} = \Gamma_i^{\text{geo}}$ $\forall i \in [\![1, 8]\!]$

3.4 Methods of the fc_simesh.siMesh object

3.4.1 fc_simesh.siMesh constructor

The constructor of the siMesh class can initialize the object from various kind of mesh file format : *.msh* (default `gmsh` format), *.mesh* (`FreeFEM++` or `Medit`) or ... (`triangle`).

Syntaxe

```
Th=fc_simesh.siMesh(meshfile)
```

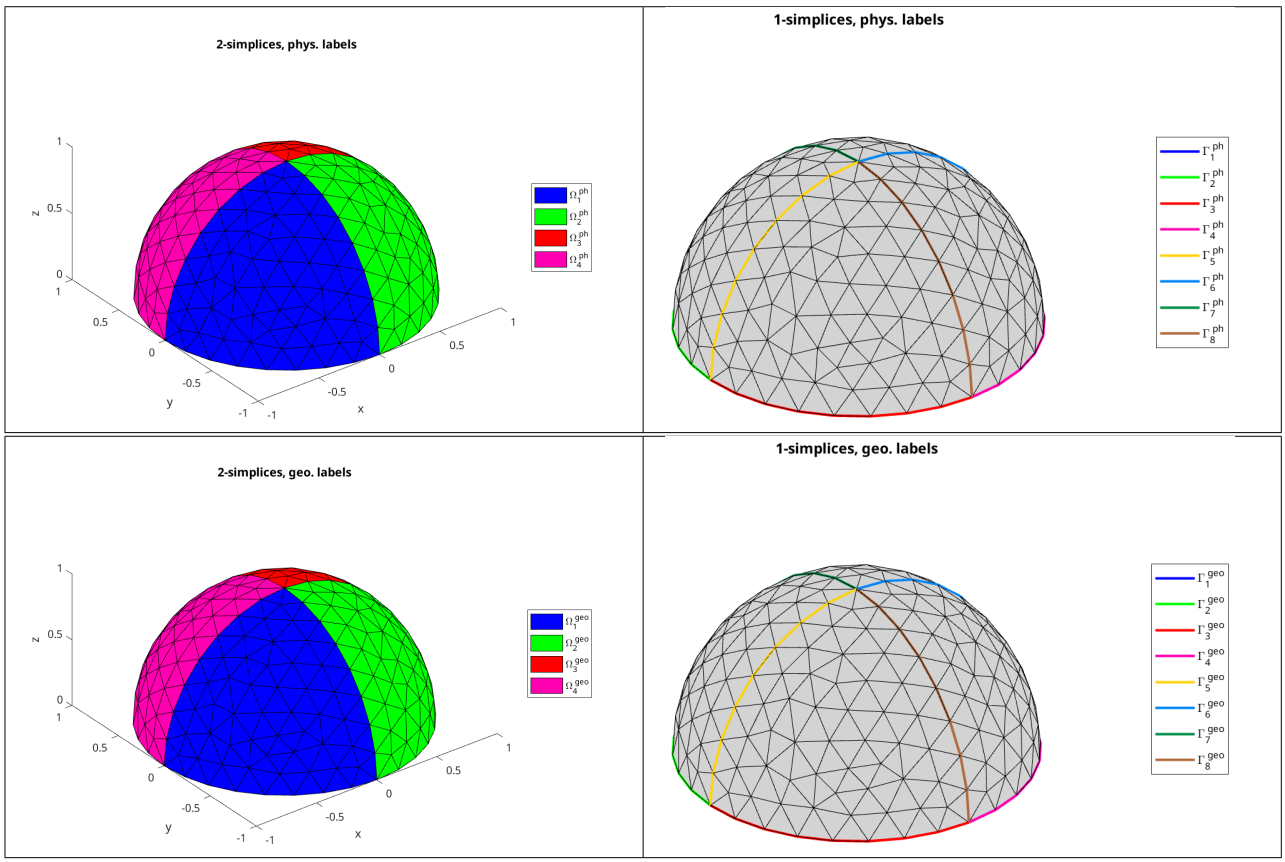


Figure 4: 3D surface Mesh from `demisphere4surf.geo`, label of the domains (left) and label of the boundaries (right)

```
Th=fc_simesh.siMesh(meshfile,Name,Value)
```

Description

Th=**fc_simesh.siMesh**(meshfile) create the siMesh object $\mathcal{T}_h$ from the mesh file meshfile (`gmsh` format by default).

Th=**fc_simesh.siMesh**(meshfile,Key,Value, ...) specifies function options using one or more Key,Value pair arguments. The string Key options can be

- `'format'` : to specify the format of the mesh file meshfile. Value must be *'medit'*, *'gmsh'* (default), *'freefem'* or *'triangle'*.
- `'dim'` : to specify the space dimension (default 2),
- `'d'` : to specify the dimensions of the simplices to read, (default [dim,dim−1])

Examples The following example use the function **fc_oogmsh.gmsh.buildmesh2d** of the fc-oogmsh toolbox to build the mesh from the *.geo* file `condenser11.geo`. This .geo file is located in the directory `geodir/2d` of the fc-oogmsh toolbox.

Listing 5: : siMesh constructor

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',25,'verbose',0);
disp('*** Read mesh ***')
Th=fc_simesh.siMesh(meshfile)
```

Output

```
*** Read mesh ***

Th =

  fc_simesh.siMesh with properties:
             d: 2 double
           dim: 2 double
           sTh: (1x46 cell)
          nsTh: 46 double
      toGlobal: (1x3162 double)
      toParent: (1x3162 double)
       sThsimp: (1x46 double)
        sThlab: (1x46 double)
     sThcolors: (46x3 double)
          bbox: [ -1 1 -1 1 ] (1x4 double)
     sThgeolab: (1x46 double)
    sThphyslab: (1x46 double)
   sThpartlabs: []
   sThboundlabs: []
            nq: 3162 double
     nqParents: 3162 double
     toParents: (1x1 cell)
         other: (1x1 struct)
```

3.4.2 find method

We denote by Th a siMesh object.

- Th.**find**(d) : returns the sorted indices array of the d-simplicial elementary meshes in the array Th.sTh.
- Th.**find**(d,labels) : returns the sorted indices of the d-simplicial elementary meshes with label in labels. labels could be an index, an array of indices. If nothing is found then return [].

Several examples are given in functions:
fc_simesh.demos.find2D(), **fc_simesh**.demos.find3D(), **fc_simesh**.demos.find3Ds()
Now some very basic samples are presented.

Listing 6: : siMesh find method samples

```
meshfile=fc_oogmsh.gmsh.buildmesh3d('quart_sphere2',5, 'verbose',0);
Th=fc_simesh.siMesh(meshfile,'dim',3);
disp(Th)
idx=Th.find(3);
fprintf('3-simplices siMeshElt \n  indices: [%s], labels=[%s]\n',num2str(idx),num2str(Th.sThlab(idx)) )
idx=Th.find(2);
fprintf('2-simplices siMeshElt \n  indices: [%s], labels=[%s]\n',num2str(idx),num2str(Th.sThlab(idx)) )
idx=Th.find(2,4);
fprintf('2-simplices siMeshElt with label==4\n  indices: [%s], ...
    labels=[%s]\n',num2str(idx),num2str(Th.sThlab(idx)) )
idx=Th.find(2,[6,4,2,10]);
fprintf('2-simplices siMeshElt with label in [6,4,2,10]\n  indices: [%s], ...
    labels=[%s]\n',num2str(idx),num2str(Th.sThlab(idx)) )
```

Output

```
  fc_simesh.siMesh with properties:
             d: 3 double
           dim: 3 double
           sTh: (1x23 cell)
          nsTh: 23 double
      toGlobal: (1x1172 double)
      toParent: (1x1172 double)
       sThsimp: [ 3 3 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 0 0 0 0 0 ] (1x23 double)
        sThlab: [ 25 27 11 13 15 17 19 21 23 1 2 3 4 5 6 7 8 9 1 2 3 4 5 ] (1x23 double)
     sThcolors: (23x3 double)
          bbox: [ -1 1 0 1 0 1 ] (1x6 double)
     sThgeolab: [ 25 27 11 13 15 17 19 21 23 1 2 3 4 5 6 7 8 9 1 2 3 4 5 ] (1x23 double)
    sThphyslab: [ 1 2 1 2 3 7 6 4 5 1 4 3 2 9 7 8 5 6 1 2 3 4 5 ] (1x23 double)
   sThpartlabs: []
   sThboundlabs: []
            nq: 1172 double
     nqParents: 1172 double
     toParents: (1x1 cell)
         other: (1x1 struct)
3-simplices siMeshElt
  indices: [1 2], labels=[25 27]
2-simplices siMeshElt
  indices: [3 4 5 6 7 8 9], labels=[11 13 15 17 19 21 23]
2-simplices siMeshElt with label==4
  indices: [], labels=[]
2-simplices siMeshElt with label in [6,4,2,10]
  indices: [], labels=[]
```

3.4.3 feval method

Evaluates a vectorized function at vertices of the mesh. We denote by Th a siMesh object.

- res=Th.**feval**(fun) : the input parameter fun is either a function or a cell array of function handles for vector-valued functions. If fun is a function then the output is an Th.nq-by-1 array. If fun is a cell array of function handles then the output is an Th.nq-by-**length**(fun) array.
- res=Th.**feval**(fun,key,value ,...) specifies function options using one or more key,value pair arguments. The string key options could be
  - `d` : to specify the d-simplicial elementary meshes on which to evaluate the function (default Th.d). A zero value is set on all vertices not in these elementary meshes.
  - `labels` : to specify the labels of the elementary meshes on which to evaluate the function (default is all). A zero value is set on all vertices not in these elementary meshes.

Several examples are given in functions:
fc_simesh.demos.feval2D01(), **fc_simesh**.demos.feval3D01(), ...
We present now some very basic samples.

Sample 1 Let $g : \mathbb{R}^2 \mapsto \mathbb{R}$ defined by $g(x, y) = \cos(x)\sin(y)$. We propose in Listing 7 four approaches to defined this function for using with **feval** method.

Listing 7: : **feval** method, four ways to defined a function

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',50,'verbose',0);
Th=fc_simesh.siMesh(meshfile);

g1=@(x,y) cos(x).*sin(y); % .* for vectorized function
g2=@(X) cos(X(1,:)).*sin(X(2,:));

z1=Th.feval(g1);
z2=Th.feval(g2);

fprintf('max(abs(z2-z1))=%e\n',max(abs(z2-z1)))
```

Output

```
max(abs(z2-z1))=0.000000e+00
```

Sample 2

Listing 8: : **feval** method with a vector-valued function

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',50,'verbose',0);
Th=fc_simesh.siMesh(meshfile)

% f : R^2 -> R^3
f={@(x,y) cos(2*x).*sin(3*y),@(x,y) cos(3*x).*sin(4*y),@(x,y) cos(4*x).*sin(5*y)};
z=Th.feval(f);
fprintf('*** nq=%d, size(z)==[%d,%d]',Th.nq,size(z))
```

Output

```
Th =

  fc_simesh.siMesh with properties:
             d: 2 double
           dim: 2 double
           sTh: (1x46 cell)
          nsTh: 46 double
      toGlobal: (1x11945 double)
      toParent: (1x11945 double)
       sThsimp: (1x46 double)
        sThlab: (1x46 double)
     sThcolors: (46x3 double)
          bbox: [ -1 1 -1 1 ] (1x4 double)
     sThgeolab: (1x46 double)
    sThphyslab: (1x46 double)
   sThpartlabs: []
  sThboundlabs: []
            nq: 11945 double
     nqParents: 11945 double
     toParents: (1x1 cell)
         other: (1x1 struct)
*** nq=11945, size(z)==[11945,3]
```

3.4.4 eval method

Evaluates numerical datas or vectorized functions at vertices of the mesh. We denote by Th a siMesh object and n_q = Th.nq the total number of vertices.

- res=Th.**eval**(data) : the input parameter data could be
  - a scalar,
  - a handle to a vectorized function,
  - a n_q-by-1 array,
  - a 1-by-m cell array of mixed previous kinds, ($m \geqslant 1$).

  The return value is a n_q-by-1 array if the input parameter data is not a cell array otherwise it's a n_q-by-m array.
- res=Th.**eval**(data,key,value,...) specifies function options using one or more key,value pair arguments. The string key options could be
  - `d` : to specify the d-simplicial elementary meshes on which to evaluate data (default Th.d). A zero value is set on all vertices not in these elementary meshes.
  - `labels` : to specify the labels of the elementary meshes on which to evaluate data (default is all). A zero value is set on all vertices not in these elementary meshes.

Several examples are given in functions:
fc_simesh.demos.eval2D01(), **siMesh**.demos.eval3D01(), ...
We present now some very basic samples.

Sample 1

Listing 9: : **eval** method, four ways to defined a function

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',50,'verbose',0);
Th=fc_simesh.siMesh(meshfile);

g1=pi*ones(Th.nq,1);
g2=pi*ones(1,Th.nq);
g3=@(X) pi;

z1=Th.eval(g1);
z2=Th.eval(g2);
z3=Th.eval(g3);

fprintf('size(z1)=[%d,%d]\n',size(z1))
fprintf('size(z2)=[%d,%d]\n',size(z2))
fprintf('size(z3)=[%d,%d]\n',size(z3))
fprintf('max(abs(z2-z1))=%e\n',max(abs(z2-z1)))
fprintf('max(abs(z3-z1))=%e\n',max(abs(z3-z1)))
```

Output

```
size(z1)=[11945,1]
size(z2)=[11945,1]
size(z3)=[11945,1]
max(abs(z2-z1))=0.000000e+00
max(abs(z3-z1))=0.000000e+00
```

Sample 2

Listing 10: : **eval** method with a vector-valued function

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',50,'verbose',0);
Th=fc_simesh.siMesh(meshfile);
u=Th.feval(@(x,y) cos(3*x).*sin(4*y));
% f : R^2 -> R^3
f={@(x,y) cos(2*x).*sin(3*y),u,@(x,y) cos(4*x).*sin(5*y),pi};
z=Th.eval(f);
fprintf('*** nq=%d, size(z)==[%d,%d]',Th.nq,size(z))
```

Output

```
*** nq=11945, size(z)==[11945,4]
```

3.4.5 get_h method

returns the maximum edges length of the mesh. We denote by Th a siMesh object.

- h=Th.get_h()

3.4.6 get_mesh method

Returns a vertices array q, a connectivity array me and a `toGlobal` indices array.

- [q,me,toGlobal]=Th.get_mesh() : returns the global vertices array q, the connectivity array me (i.e. all the Th.d-simplices of the mesh). In this case, toGlobal is just 1:Th.nq.
- [q,me,toGlobal]=Th.get_mesh(key,value,...) specifies function options using one or more key,value pair arguments. The string key options could be
  - 'd' : to specify the d-simplicial elementary meshes to consider.
  - 'labels' : to specify the labels of the elementary meshes to consider.

  In this case, toGlobal is a 1-by-**length**(q) array (subset of 1:Th.nq). If we denote by qglob the global vertices array then
  qglob(:, toGlobal)==q

Several examples are given in functions:
fc_simesh.demos.get_mesh2D(), **siMesh**.demos.get_mesh3D(), **siMesh**.demos.get_mesh3Ds()

Listing 11: : get_mesh method, four ways to defined a function

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',50,'verbose',0);
Th=fc_simesh.siMesh(meshfile);

[q,me,toGlobal]=Th.get_mesh();
[q2,me2,toGlobal2]=Th.get_mesh('labels',2:2:8);
[q1,me1,toGlobal1]=Th.get_mesh('d',1,'labels',1:8);

fprintf('norm(q(:,toGlobal2)-q2,Inf)=%e\n',norm(q(:,toGlobal2)-q2,Inf))
fprintf('norm(q(:,toGlobal1)-q1,Inf)=%e\n',norm(q(:,toGlobal1)-q1,Inf))
```

Output

```
norm(q(:,toGlobal2)-q2,Inf)=0.000000e+00
norm(q(:,toGlobal1)-q1,Inf)=0.000000e+00
```

3.4.7 get_nme method

Returns the number of d-simplicial elements with $d = \mathcal{T}_h$.d by default. We denote by Th a siMesh object.

- nme=Th.get_nme() : returns the number of Th.d-simplicial elements in the mesh.
- nme=Th.get_mesh(key,value,...) specifies function options using one or more key,value pair arguments. The string key options could be
  - 'd' : to specify the d-simplicial elementary meshes to consider.
  - 'labels' : to specify the labels of the elementary meshes to consider.

Listing 12: : get_nme method

```
meshfile=fc_oogmsh.gmsh.buildmesh3d('quart_sphere2',5);
Th=fc_simesh.siMesh(meshfile);
for d=[Th.d:-1:0]
  fprintf('Number of %d-simplices : %d\n',d,Th.get_nme('d',d))
end

nme=Th.get_nme('d',2,'labels',1:4);
fprintf('Number of 2-simplices in union of label''s 1 to 4 : %d\n',nme);
```

Output

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/3d/quart_sphere2.geo
[fc-oogmsh] Mesh file <fc-oogmsh>/meshes/quart_sphere2-5.msh [version 4.1] already exists.
  -> Use "force" flag to rebuild if needed.
Number of 3-simplices : 4778
Number of 2-simplices : 1653
Number of 1-simplices : 115
Number of 0-simplices : 5
Number of 2-simplices in union of label's 1 to 4 : 0
```

3.4.8 get_nq method

Returns the number of vertices in the union of some elementary meshes. By default all the (Th.d)-simplicial elementary meshes are selected. We denote by Th a siMesh object.

- nq=Th.get_nq() : returns the number of vertices in the union of the Th.d-simplicial elementary meshes.

- nq=Th.get_nq(key,value,...) specifies function options using one or more key,value pair arguments. The string key options could be
  - 'd' : to specify the d-simplicial elementary meshes to consider.
  - 'labels' : to specify the labels of the elementary meshes to consider.

Listing 13: : get_nqe method

```
meshfile=fc_oogmsh.gmsh.buildmesh3d('quart_sphere2',5);
Th=fc_simesh.siMesh(meshfile);
for d=[Th.d:-1:0]
  fprintf('Number of vertices in %d-simplices elementary meshes : %d\n',d,Th.get_nq('d',d))
end

nq=Th.get_nq('d',2,'labels',1:4);
fprintf('Number of vertices in the union of 2-simplices elementary meshes of label''s 1 to 4 : %d\n',nq);
```

Output

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/3d/quart_sphere2.geo
[fc-oogmsh] Mesh file <fc-oogmsh>/meshes/quart_sphere2-5.msh [version 4.1] already exists.
  -> Use "force" flag to rebuild if needed.
Number of vertices in 3-simplices elementary meshes : 1172
Number of vertices in 2-simplices elementary meshes : 812
Number of vertices in 1-simplices elementary meshes : 111
Number of vertices in 0-simplices elementary meshes : 5
Number of vertices in the union of 2-simplices elementary meshes of label's 1 to 4 : 0
```

3.4.9 get_labels method

Returns the labels of the d-simplicial elementary meshes. We denote by Th a siMesh object.

- labels=Th.get_labels(d) : the labels of the d-simplicial elementary meshes.

Listing 14: : get_labels method

```
geofile=fc_simesh.get_geo(3,3,'quart_sphere2');
meshfile=fc_oogmsh.gmsh.buildmesh3d(geofile,10);
Th=fc_simesh.siMesh(meshfile);
lab3=Th.get_labels(3)
lab2=Th.get_labels(2)
lab1=Th.get_labels(1)
lab0=Th.get_labels(0)
```

Output

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/3d/quart_sphere2.geo
[fc-oogmsh] Mesh file <fc-oogmsh>/meshes/quart_sphere2-10.msh [version 4.1] already exists.
  -> Use "force" flag to rebuild if needed.

lab3 =

   25   27


lab2 =

   11   13   15   17   19   21   23


lab1 =

    1    2    3    4    5    6    7    8    9


lab0 =

    1    2    3    4    5
```

We can refer to Figure 3 to validate the output results.

3.4.10 move method

Moving a mesh. We denote by Th a siMesh object.

- Th.move(u,dims) : the input parameter u is the displacement vector which is either a numerical array or a cell array of numerical array. The second parameter dims is uses to specify the dimensions where displacement vector is applied.
  - Let U=u if u is an n-by-n_q array and U=u.' if u is an n_q-by-n array. Then dims is a vector of length n and all nodes array q in Th are replaced by

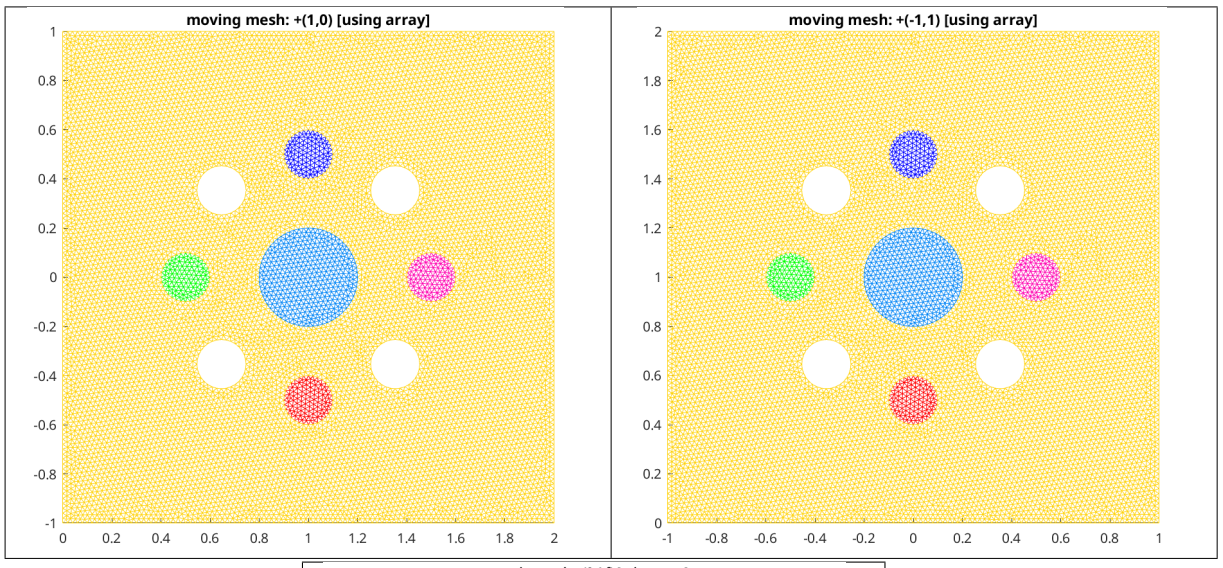
    Th.sThi.q(dims,:)=q(dims,:)+U(:,Th.sThi.toGlobal))

– u is a cell array of length d. Let U=u{k} if u{k} is an 1-by-n_q array and U=u{k}.' if fcmcodeu{i} is an n_q-by-1 array.

Th.sThi.q(dims(k),:)=q(dims(k),:)+U(:,Th.sThi.toGlobal))

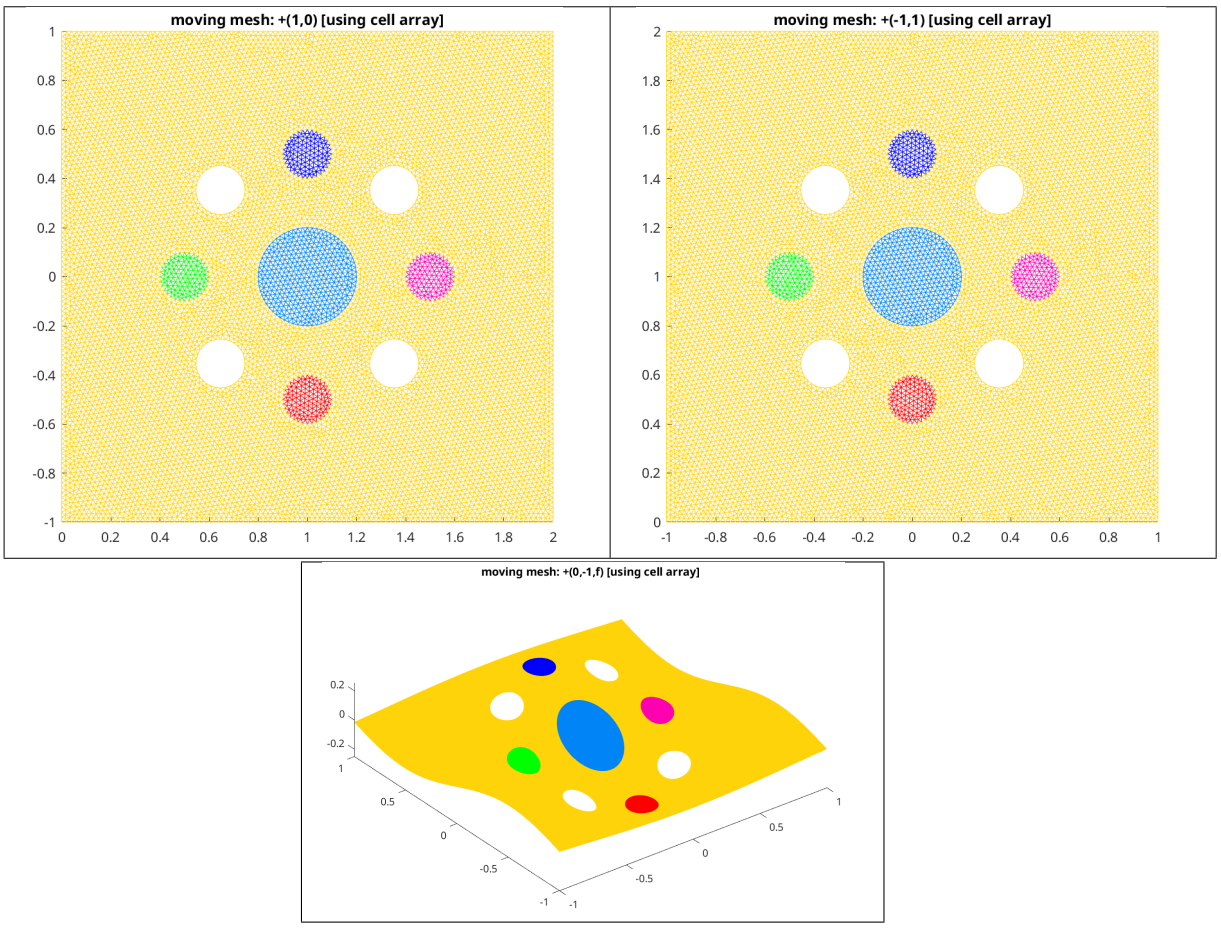
remark 3.1

1. Take care that modification in Th are done *inplace*. One can use the command Th1=Th.copy() which make a deep copy of the Th object (modifying one object not change its deep copy).
2. If **max**(dims) greather than Th.dim, then the mesh dimension is automaticaly increased and new dimensions in nodes array are set to zeros.



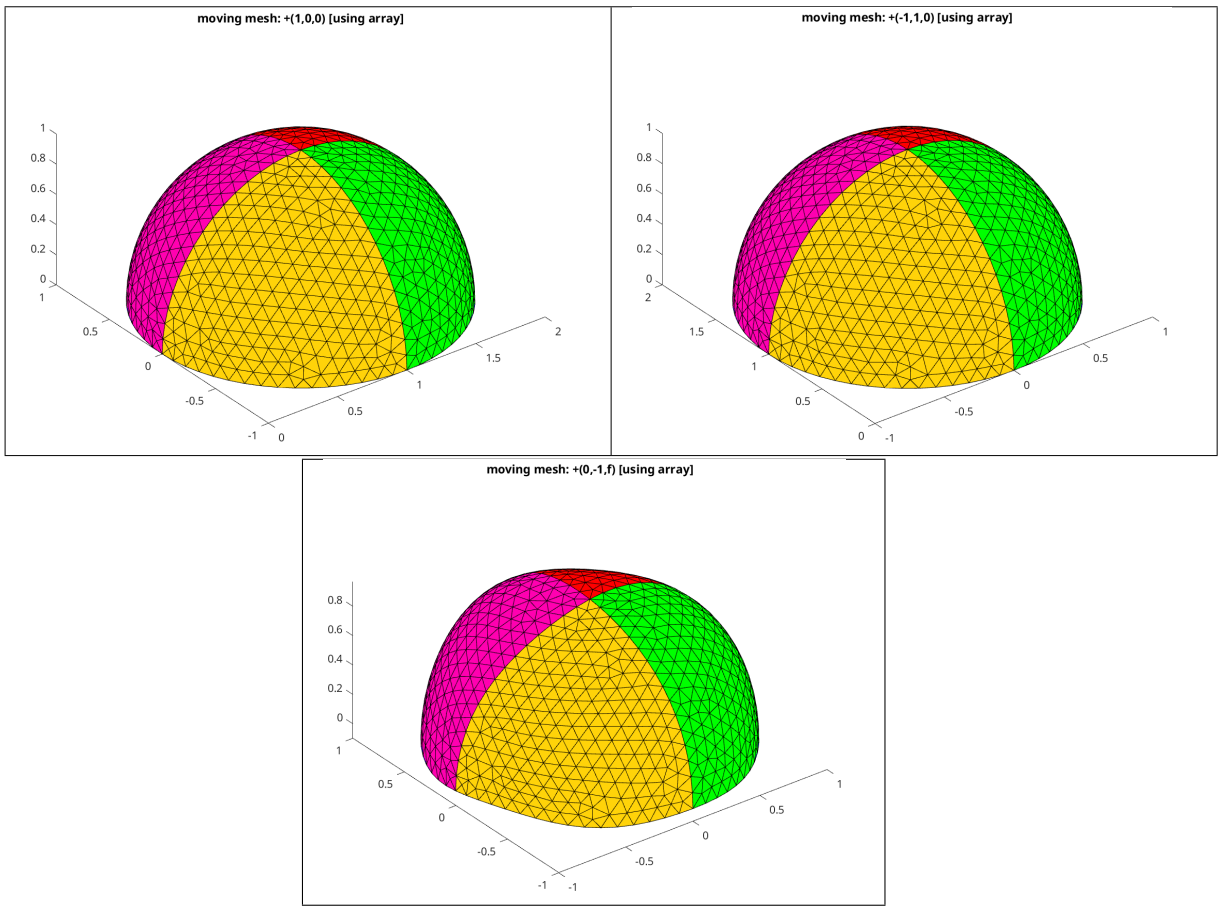

```
[Th,verbose]=fc_simesh.demos.setMesh2D(varargin{:});
Th1=Th.copy(); % Deep copy of Th
f=@(x,y) cos(2*x).*sin(3*y)/4;
U=ones(1,Th.nq);

Th1.move(U,1)
figure(1)
Th1.plotmesh();
axis image; title('moving mesh: +(1,0) [using array]')

Th1.move([-U;U],1:2)
figure(2)
Th1.plotmesh();
axis image; title('moving mesh: +(-1,1) [using array]')

Th1.move([U;Th.eval(f)]', 2:3) % [U;Th.eval(f)] OK
figure(3)
Th1.plotmesh('edgecolor','none');
axis image; title('moving mesh: +(0,1,f) [using array]')
```

Listing 15: Using the fc_simesh.moveArray function with a 2D mesh, part of the fc_simesh.demos.moveArray2D function



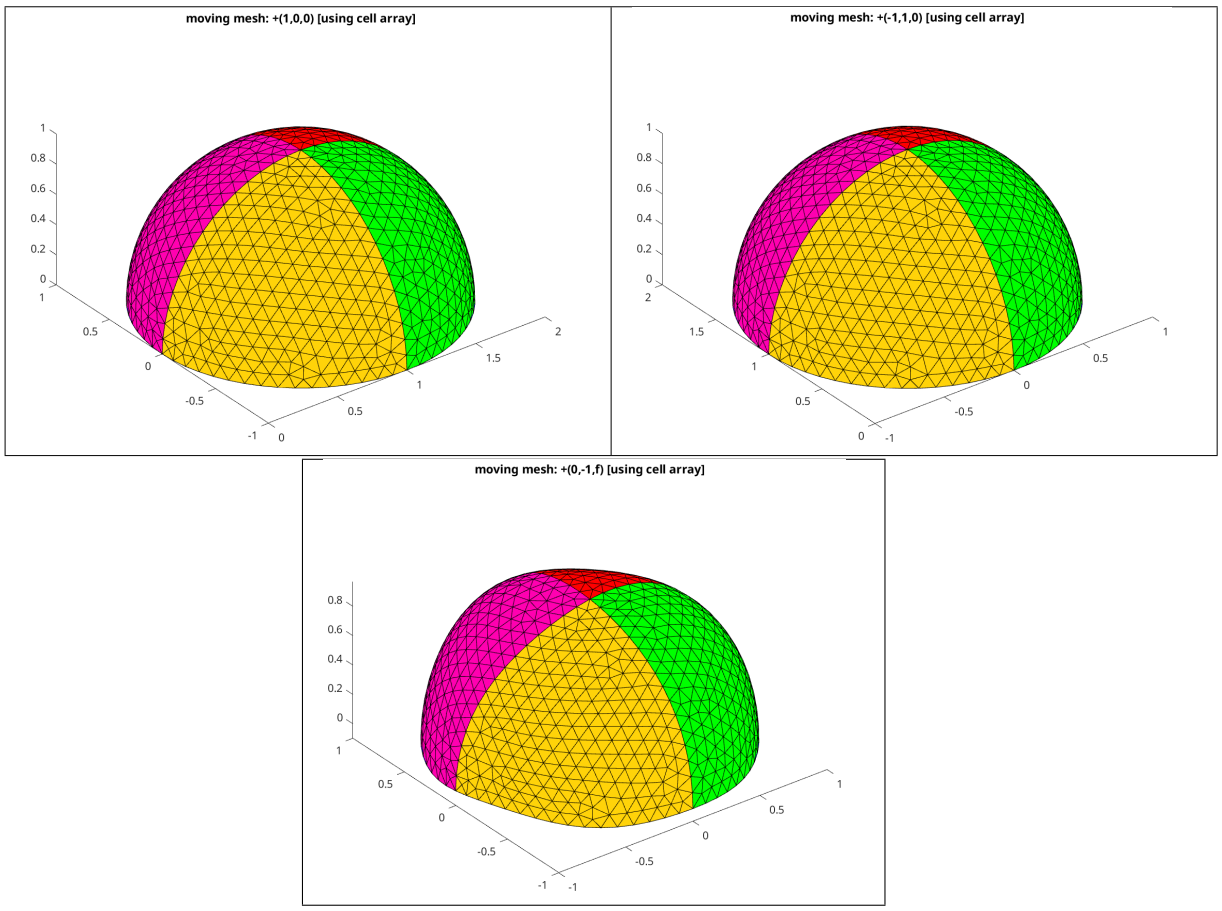

```
[Th,verbose]=fc_simesh.demos.setMesh2D(varargin{:});
Th1=Th.copy(); % Deep copy of Th
f=@(x,y) cos(2*x).*sin(3*y)/4;
U=ones(1,Th.nq);

Th1.move({U}, 1) % Th1.move(U, 1) also is OK!
figure(1)
Th1.plotmesh();
axis image; title('moving mesh: +(1,0) [using cell array]')

Th1.move({-U,U}, 1:2)
figure(2)
Th1.plotmesh();
axis image; title('moving mesh: +(-1,1) [using cell array]')

Th1.move({-U;Th.eval(f)},2:3) % Th1 is now a 3D mesh
figure(3)
Th1.plotmesh('edgecolor','none');
axis image; title('moving mesh: +(0,-1,f) [using cell array]')
```

Listing 16: Using the fc_simesh.moveCell function with a 2D mesh, part of the fc_simesh.demos.moveCell2D function



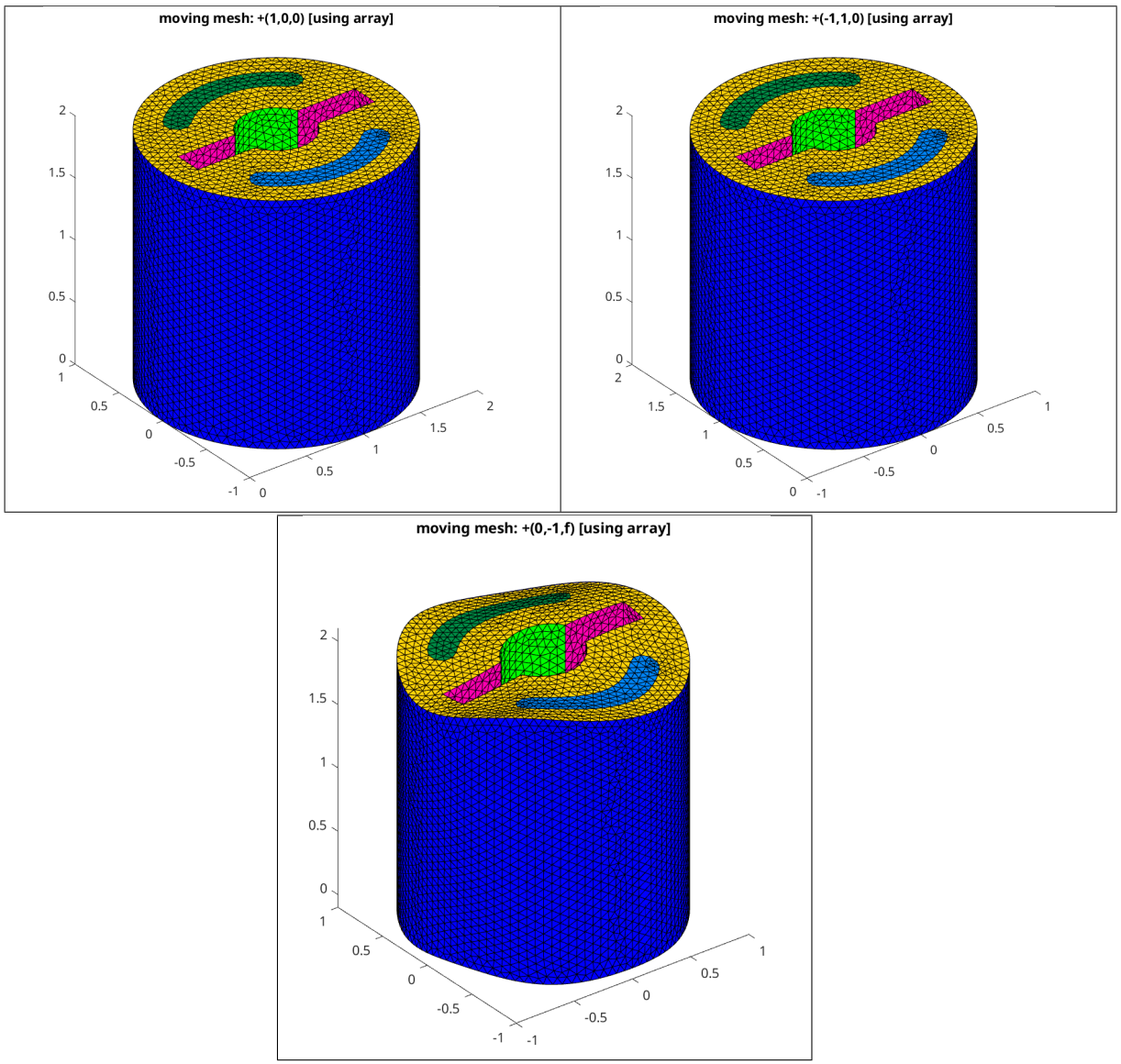

```
[Th,verbose]=fc_simesh.demos.setMesh3Ds(varargin{:});
Th1=Th.copy(); % Deep copy of Th
f=@(x,y,z) cos(2*x+z).*sin(3*y-z)/10;
U=ones(1,Th.nq);

Th1.move(U,1)
figure(1)
Th1.plotmesh();
axis image; title('moving mesh: +(1,0,0) [using array]')

Th1.move([-U;U], 1:2)
figure(2)
Th1.plotmesh();
axis image; title('moving mesh: +(-1,1,0) [using array]')

Th1.move([-U;Th.eval(f)'], 2:3)
figure(3)
Th1.plotmesh();
axis image; title('moving mesh: +(0,-1,f) [using array]')
```

Listing 17: Using the **fc_simesh.moveArray** function with a 3Ds mesh, part of the **fc_simesh.demos.moveArray3Ds** function



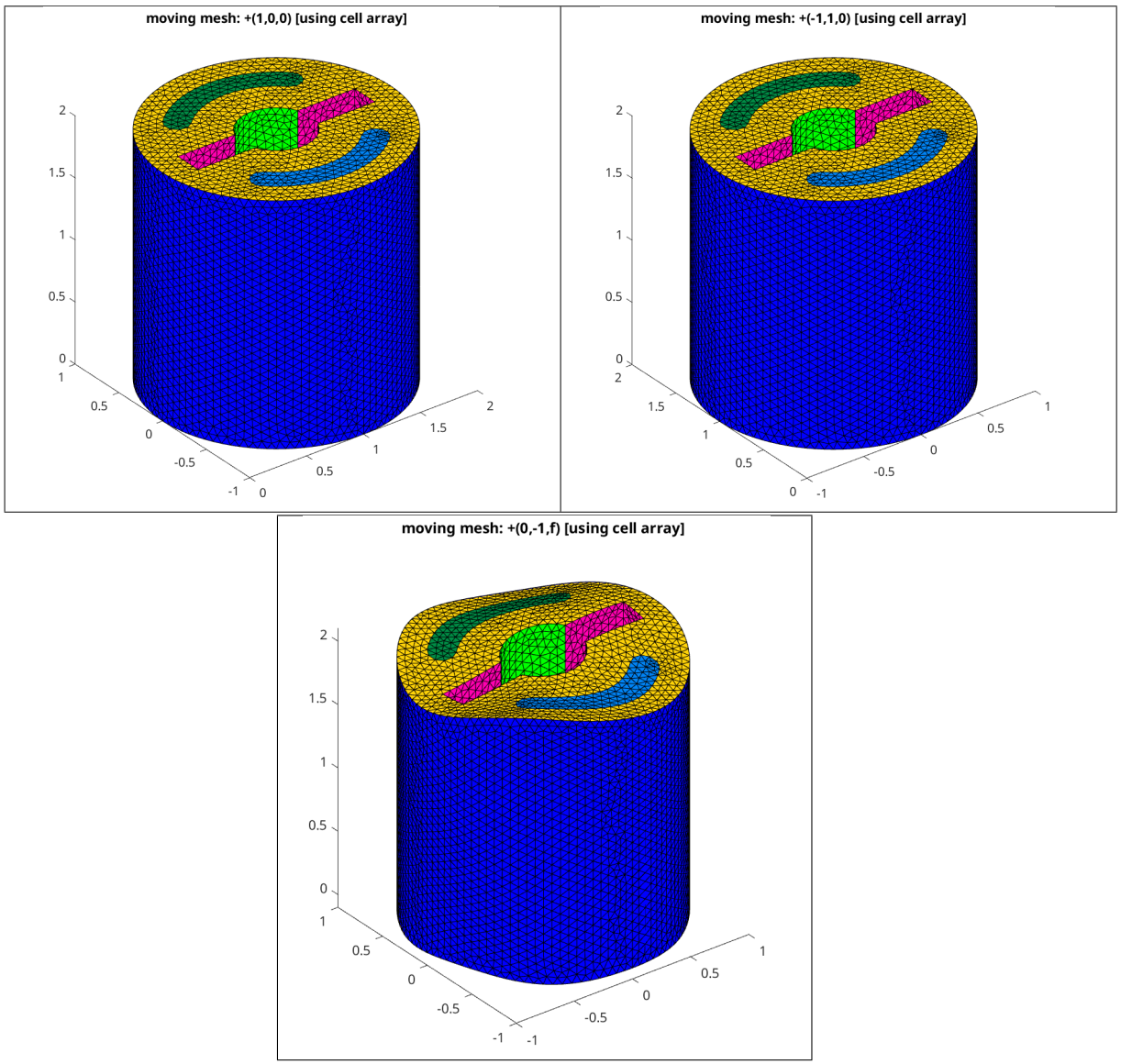

```
[Th,verbose]=fc_simesh.demos.setMesh3Ds(varargin{:});
Th1=Th.copy(); % Deep copy of Th
f=@(x,y,z) cos(2*x+z).*sin(3*y-z)/10;
U=ones(1,Th.nq);

Th1.move({U},1)
figure(1)
Th1.plotmesh();
axis image;title('moving mesh: +(1,0,0) [using cell array]')

Th1.move({-U,U},1:2)
figure(2)
Th1.plotmesh();
axis image;title('moving mesh: +(-1,1,0) [using cell array]')

Th1.move({-U;Th.eval(f)}, 2:3)
figure(3)
Th1.plotmesh();
axis image;title('moving mesh: +(0,-1,f) [using cell array]')
```

Listing 18: Using the fc_simesh.moveCell function with a 3Ds mesh, part of the fc_simesh.demos.moveCell3Ds function




```
Th1=Th.copy(); % Deep copy of Th
f=@(x,y,z) cos(2*x+z).*sin(3*y-z)/10;
U=ones(1,Th.nq);

Th1.move(U,1)
figure(1)
Th1.plotmesh('d',2);
axis image; title('moving mesh: +(1,0,0) [using array]')

Th1.move([-U;U], 1:2)
figure(2)
Th1.plotmesh('d',2);
axis image; title('moving mesh: +(-1,1,0) [using array]')

Th1.move([-U',Th.eval(f)]', 2:3) % [-U',Th.eval(f)] is also OK
figure(3)
Th1.plotmesh('d',2);
```

Listing 19: Using the **fc_simesh.moveArray** function with a 3D mesh, part of the **fc_simesh.demos.moveArray3D** function




```
[Th,verbose]=fc_simesh.demos.setMesh3D(varargin{:});
Th1=Th.copy(); % Deep copy of Th
f=@(x,y,z) cos(2*x+z).*sin(3*y-z)/10;
U=ones(1,Th.nq);

Th1.move({U},1)
figure(1)
Th1.plotmesh('d',2);
axis image; title('moving mesh: +(1,0,0) [using cell array]')

Th1.move({-U,U}, 1:2)
figure(2)
Th1.plotmesh('d',2);
axis image; title('moving mesh: +(-1,1,0) [using cell array]')

Th1.move({-U;Th.eval(f)}, 2:3)
figure(3)
Th1.plotmesh('d',2);
axis image; title('moving mesh: +(0,-1,f) [using cell array]')
```

Listing 20: Using the fc_simesh.moveCell function with a 3D mesh, part of the fc_simesh.demos.moveCell3D function

3.4.11 plotmesh method

The **plotmesh** method displays the mesh or parts of the mesh defined by an siMesh object.

Syntaxe

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

Description

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

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

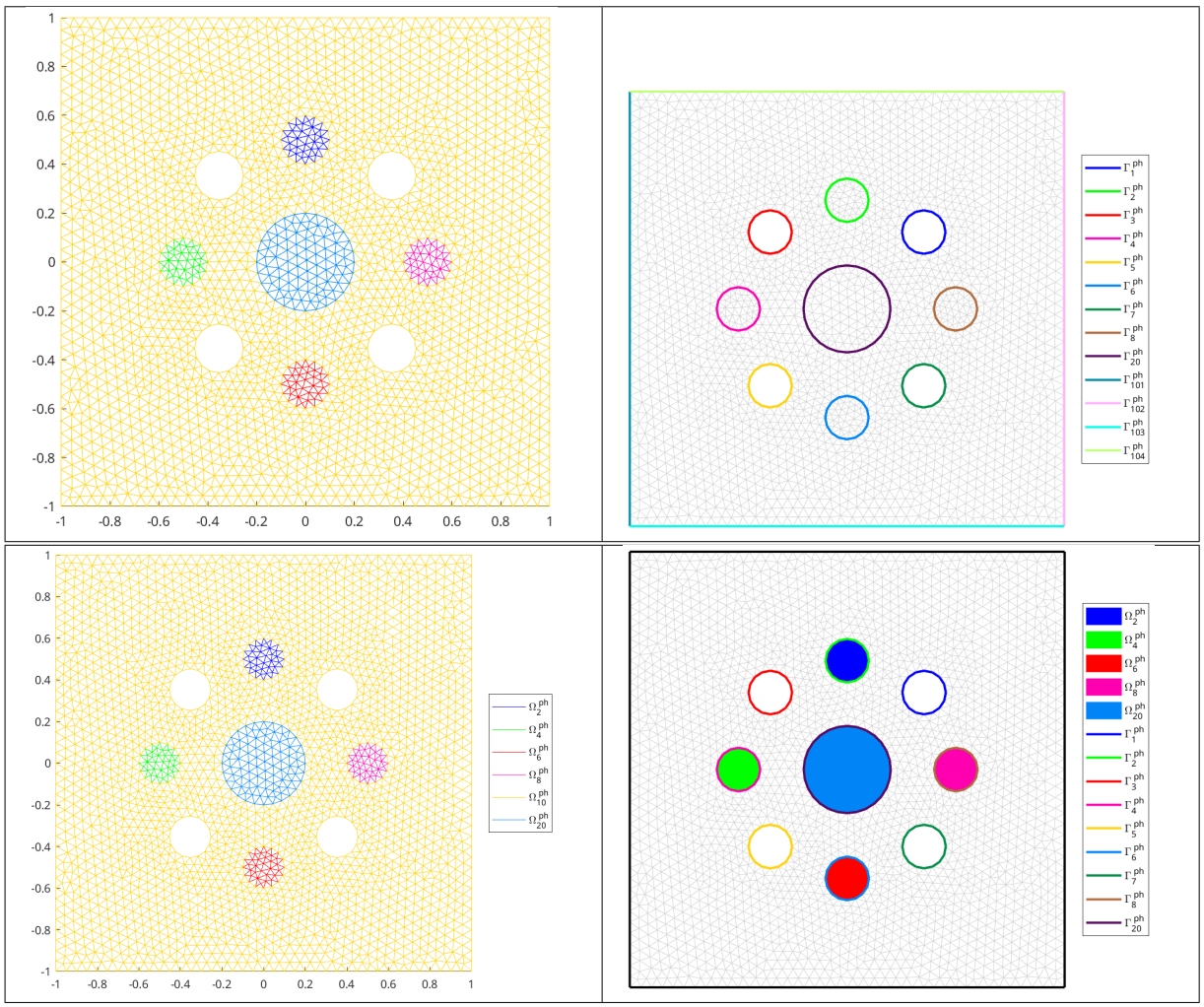
- 'd' : to specify the dimension of the simplices elements (default : Th.d)
- 'labels' : to select the labels of the elements to display,
- 'color' : to specify the color of the displayed mesh elements. (default : use one color by displayed mesh elements),
- 'inlegend' : add a legend name to graph if 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).

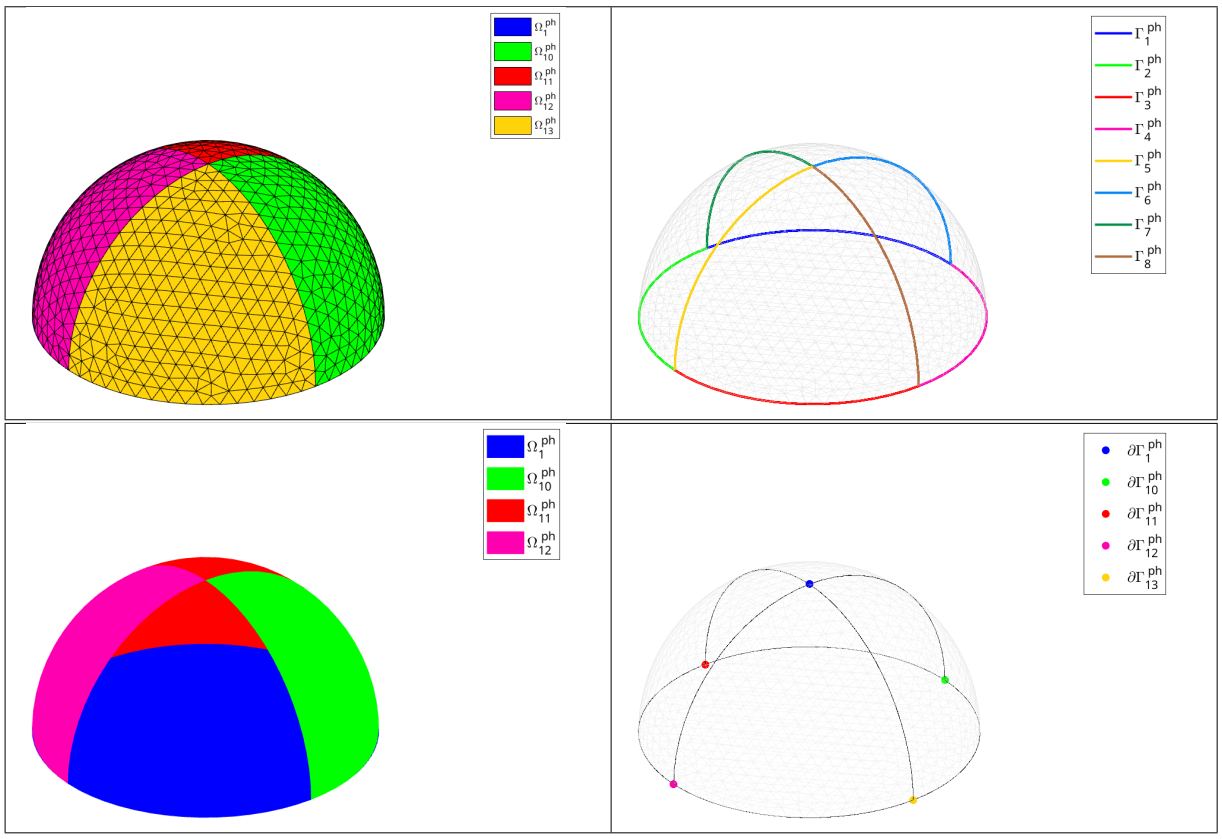
The options of second level depend on the type of elementaries mesh elements to represent.
One can use any option of the following functions according to the type of d-simplex to be represented.

- 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,



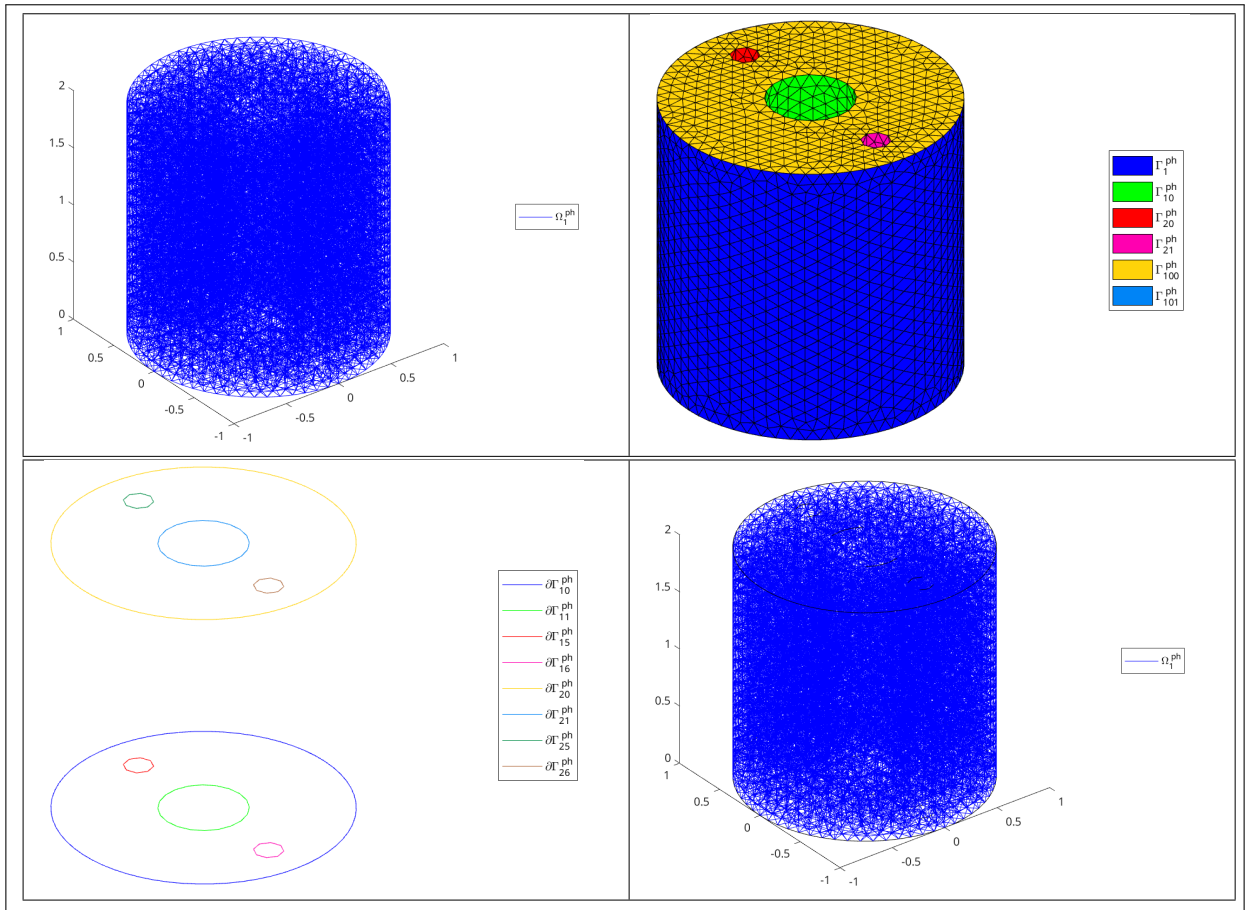
```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',20,varargin{:});
Th=fc_simesh.siMesh(meshfile);
fc_tools.graphics.monitors.onGrid(2,2);
figure(1)
Th.plotmesh();
axis image

figure(2)
Th.plotmesh('color','LightGray')
hold on;axis image;axis off
Th.plotmesh('d',1,'Linewidth',2,'inlegend',true);
fc_graphics4mesh.legend('Location','eastoutside')

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

figure(4)
Th.plotmesh('color','LightGray','labels',10)
hold on;axis image;axis off
Th.plotmesh('fill',true,'labels',[2:2:8,20],'inlegend',true)
Th.plotmesh('d',1,'Linewidth',2,'inlegend',true,'labels',[1:8,20]);
Th.plotmesh('d',1,'Linewidth',2,'color','k','labels',[101:104]);
fc_graphics4mesh.legend('Location','eastoutside')
```

Listing 21: Using the **fc_simesh.plotmesh** function with a 2D mesh, part of the **fc_simesh.demos.plotmesh2D** function




```
meshfile=fc_oogmsh.buildmesh3ds('demisphere5',10,varargin{:});
Th=fc_simesh.siMesh(meshfile);
fc_tools.graphics.monitors.onGrid(2,2);
figure(1)
Th.plotmesh('inlegend',true)
axis off;axis equal;
fc_graphics4mesh.legend()

figure(2)
Th.plotmesh('EdgeColor','LightGray', 'EdgeAlpha',0.4,'FaceColor','none')
view(3);hold on;axis off;axis equal
Th.plotmesh('d',1,'inlegend',true,'LineWidth',2)
fc_graphics4mesh.legend()
set(legend(),'Location','NorthEastOutside','FontSize',12)

figure(3)
Th.plotmesh('labels',[1,10,11,12],'EdgeColor','none','inlegend',true)
fc_graphics4mesh.legend('Location','NorthEastOutside','FontSize',12)
axis off;axis equal

figure(4)
Th.plotmesh('EdgeColor',0.9*[1,1,1], 'EdgeAlpha',0.4,'FaceColor','none')
hold on;axis off;axis equal
Th.plotmesh('d',1,'color','k')
Th.plotmesh('d',0,'inlegend',true)
fc_graphics4mesh.legend('Location','NorthEastOutside','FontSize',12)
```

Listing 22: Using the **fc_simesh.plotmesh** function with a 3Ds mesh, part of the **fc_simesh.demos.plotmesh3Ds** 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)
Th.plotmesh('inlegend',true)
axis image
fc_graphics4mesh.legend('Location','EastOutside')

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

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

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

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

figure(6)
Th.plotmesh('d',2,'edgecolor',0.8*[1 1 1],'facecolor','None','edgealpha',0.5)
hold on;axis image
Th.plotmesh('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])];
Th.plotmesh('cutPlane',P,'Color','DarkGrey')
hold on;axis image
Th.plotmesh('d',2,'cutPlane',P,'inlegend',true);
fc_graphics4mesh.legend('Location','EastOutside')
```

Listing 23: Using the fc_simesh.plotmesh function with a 3D mesh, part of the fc_simesh.demos.plotmesh3D function

3.4.12 plot method

The **plot** method displays scalar datas on the mesh or parts of the mesh defined by an siMesh object. We denote by Th a siMesh object.

Syntaxe

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

Description

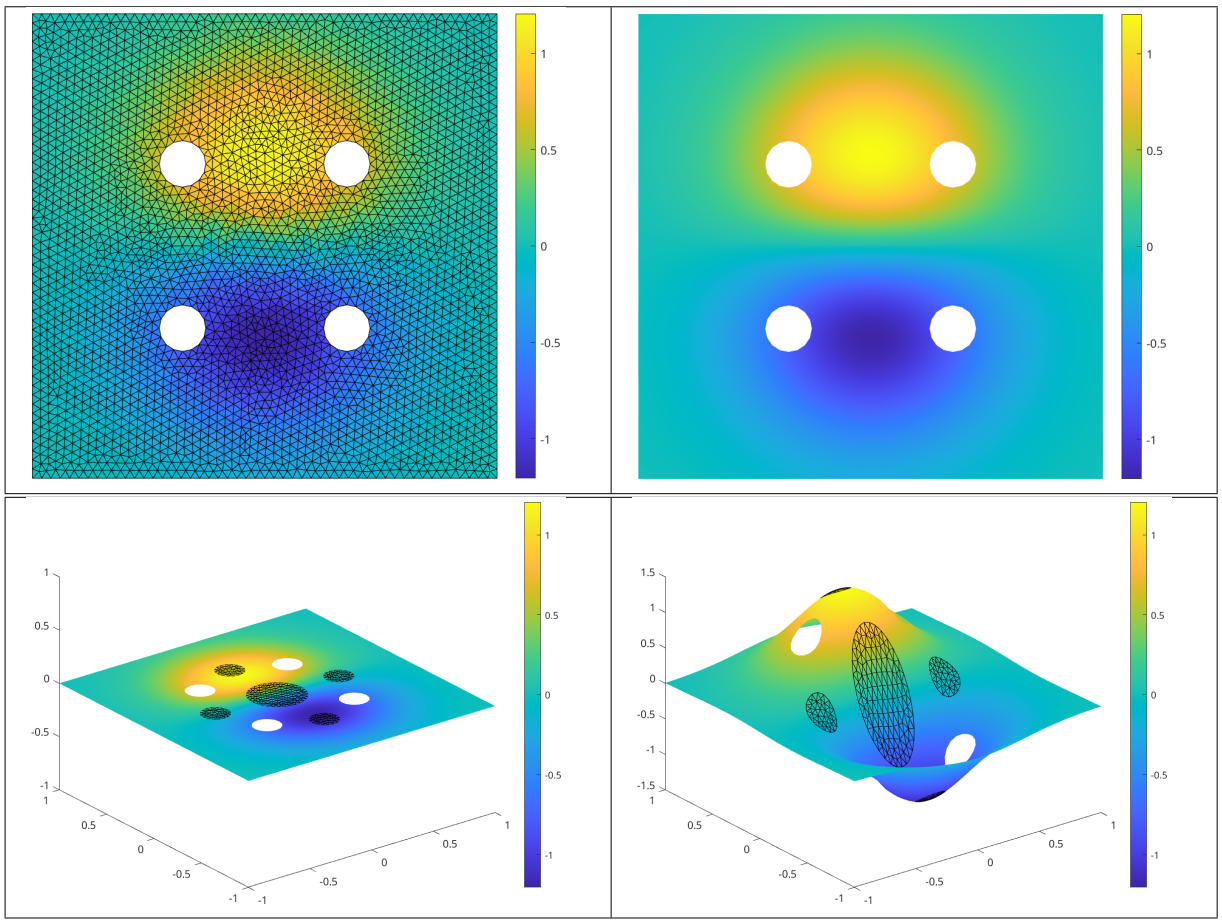
Th.**plot**(u) displays data u on all the (Th.d)-dimensional simplices elements of Th, a siMesh object. The data u is an 1D-array of size Th.nq or Th.nqGlobal or Th.nqParent.

Th.**plot**(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**)

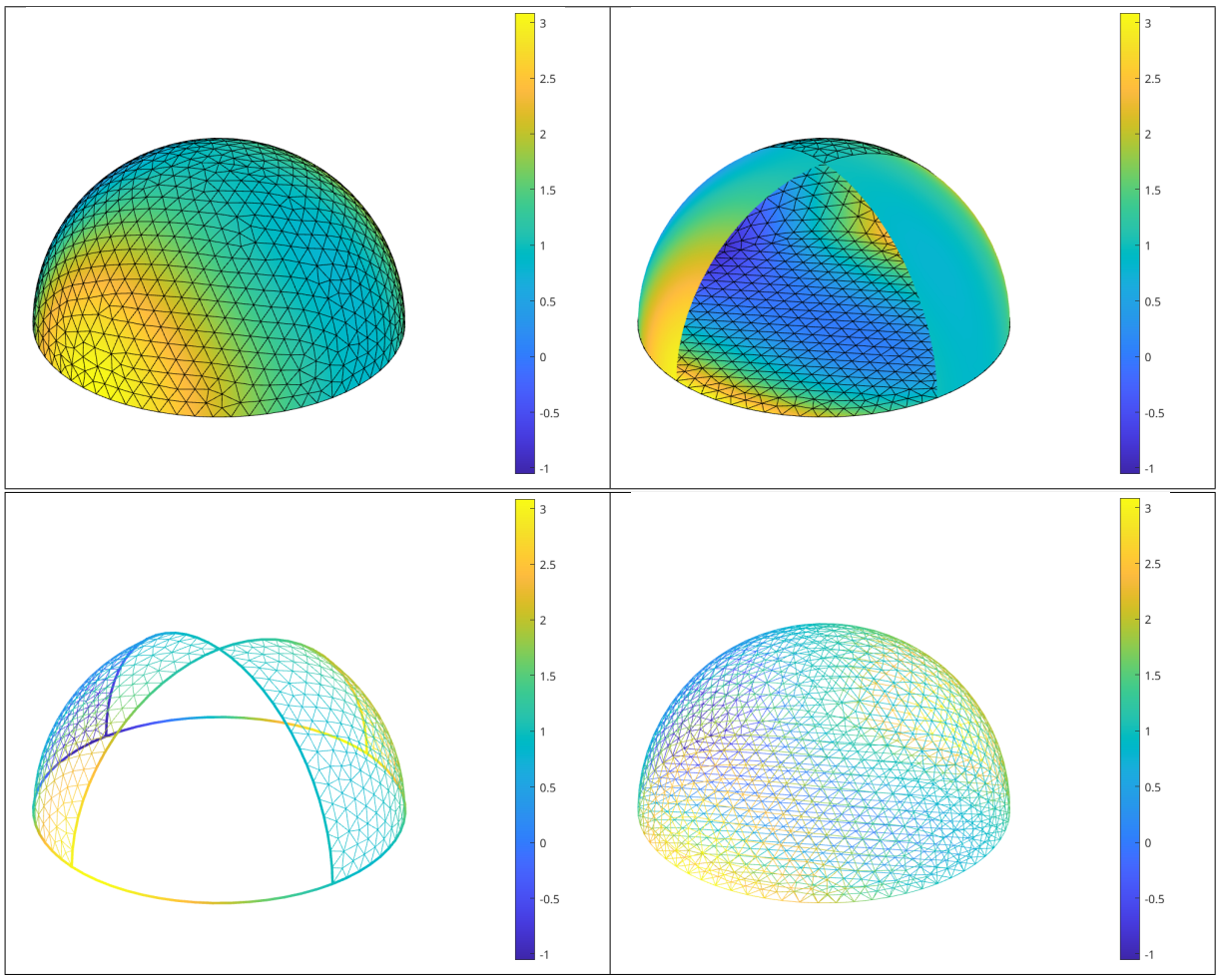
The options of second level depend on the type of elementaries mesh elements on which we want to represent datas.
One can use any option of the following functions according to the type of d-simplex.

- In dimension 3, **patch** function is used for $d \in [\![1, 3]\!]$.
- 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



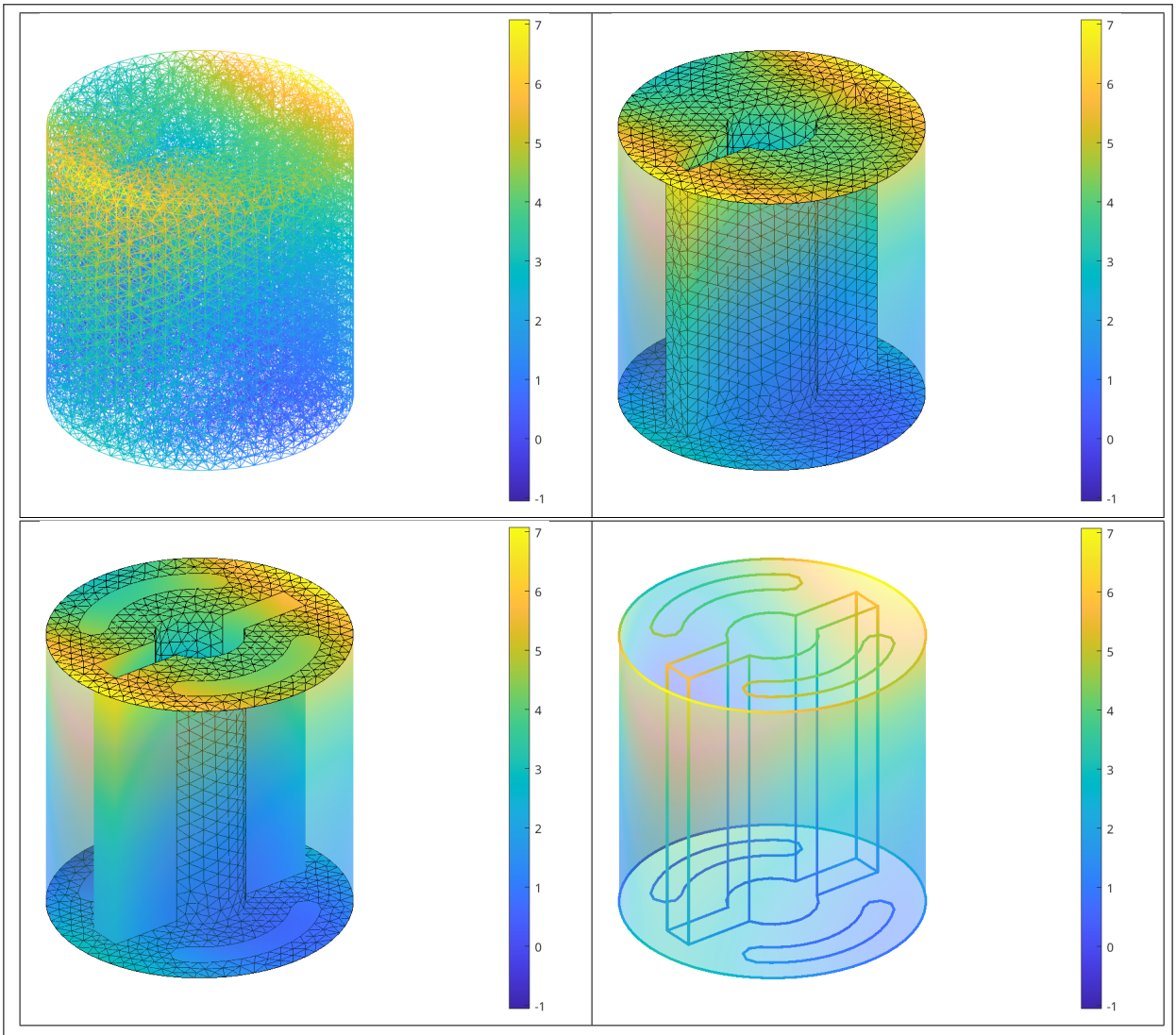
```
geofile=fc_simesh.get_geo(2,2,'condenser11');
meshfile=fc_oogmsh.gmsh.buildmesh2d(geofile,25,'verbose',0,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=Th.eval(@(x,y) 5*exp(-3*(x.^2+y.^2)).*cos(x).*sin(y));
figure(1)
Th.plot(u)
axis off;axis image;colorbar

figure(2)
Th.plot(u)
axis off;axis image;shading interp;colorbar

figure(3)
Th.plot(u,'labels',[2:2:8,20],'FaceColor','interp','plane',true)
view(3);hold on;colorbar
Th.plot(u,'labels',10,'FaceColor','interp','EdgeColor','none','plane',true)

figure(4)
Th.plot(u,'labels',[2:2:8,20],'FaceColor','interp')
view(3);hold on;colorbar
Th.plot(u,'labels',10,'FaceColor','interp','EdgeColor','none')
```

Listing 24: Using the fc_simesh.plot function with a 2D mesh, part of the fc_simesh.demos.plot2D 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);
figure(1)
Th.plot(u)
axis off;axis image;colorbar;
figure(2)
Th.plot(u,'labels',[1,11])
hold on;axis off;axis image;colorbar;
Th.plot(u,'labels',[10,12], 'FaceColor','interp', 'EdgeColor','none')
figure(3)
Th.plot(u,'d',1,'LineWidth',2)
hold on;axis off;axis image;colorbar
Th.plot(u,'labels',[10,12], 'FaceColor','none', 'EdgeColor','interp')
figure(4)
Th.plot(u,'FaceColor','none', 'EdgeColor','interp')
axis off;axis image;colorbar;
```

Listing 25: Using the fc_simesh.plot function with a 3Ds mesh, part of the fc_simesh.demos.plot3Ds function



```
geofile=fc_simesh.get_geo(3,3,'cylinderkey03');
meshfile=fc_oogmsh.buildmesh3d(geofile,10,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=Th.eval(@(x,y,z) 3*x.^2-y.^3+z.^2+x.*y);
figure(1)
Th.plot(u);
axis off;axis image;colorbar
figure(2)
Th.plot(u,'d',2,'labels',[10,11,31,1000,1020,1021,2000,2020,2021])
hold on;axis off;axis image;colorbar
Th.plot(u,'d',2,'labels',1,'FaceColor','interp', 'EdgeColor','none','FaceAlpha',0.4)
figure(3)
Th.plot(u,'d',2,'labels',1,'FaceColor','interp', 'EdgeColor','none','FaceAlpha',0.4)
hold on;axis off;axis image;colorbar
Th.plot(u,'d',2,'labels',[10,11,1000,2000])
Th.plot(u,'d',2,'labels',[31,1020,1021,2020,2021], 'FaceColor','interp','EdgeColor','none')
figure(4)
Th.plot(u,'d',2,'labels',1,'FaceColor','interp', 'EdgeColor','none','FaceAlpha',0.4)
hold on;axis off;axis image;colorbar
Th.plot(u,'d',1,'LineWidth',2)
```

Listing 26: Using the fc_simesh.plot function with a 3D mesh, part of the fc_simesh.demos.plot3D function

3.4.13 plotiso method

The plotiso method 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

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

Description

Th.**plotiso**(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.

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

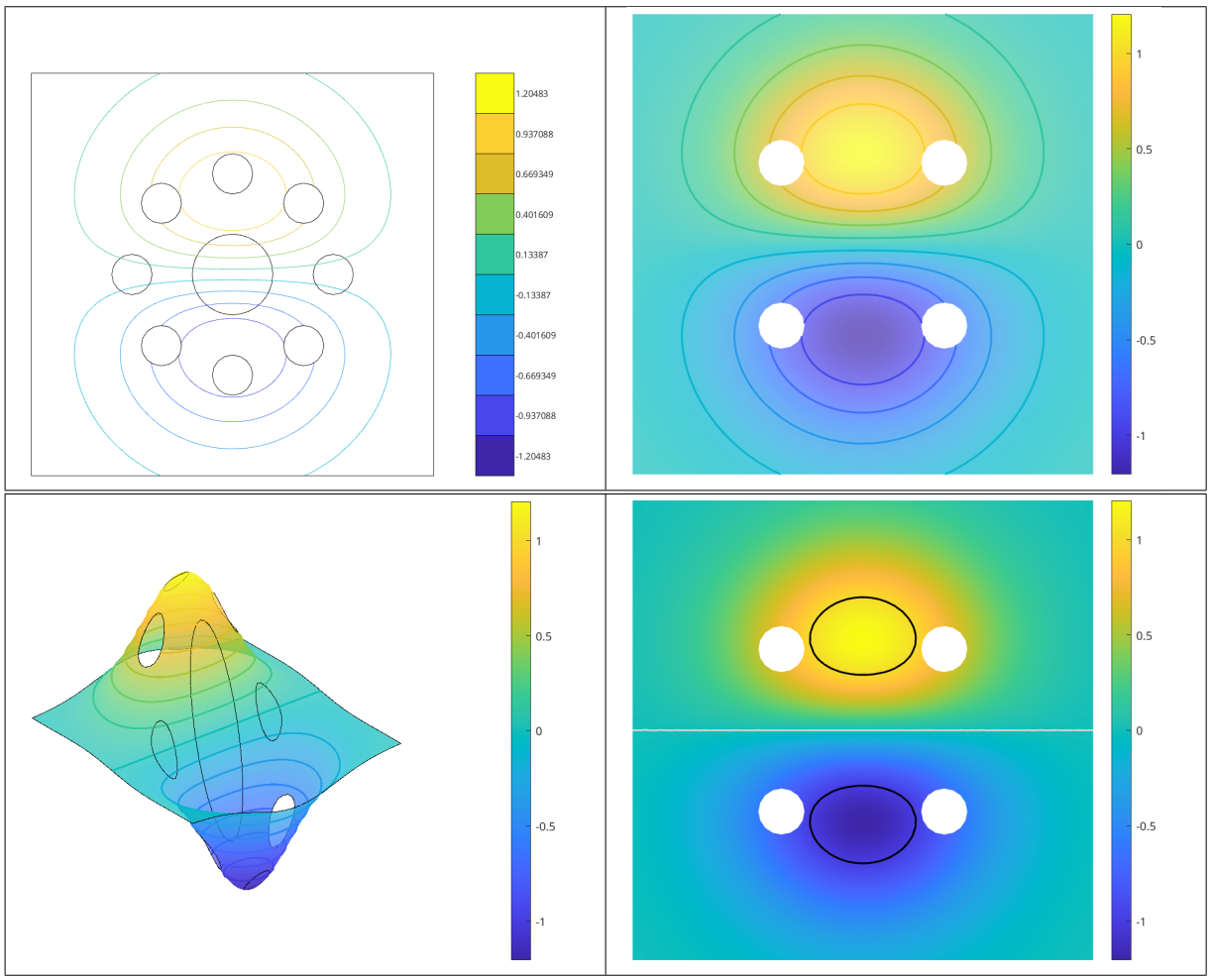
- 'niso' : to specify the number of isolines (default : 10)
- 'isorange' : to specify the list of isovalues (default : empty)
- 'isocolorbar' : if **true**, colorbar with isovalues is drawn (default : **false**)
- 'format' : to specify the format of the isovalues on the colorbar (default : '%g')
- 'labels' : 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**

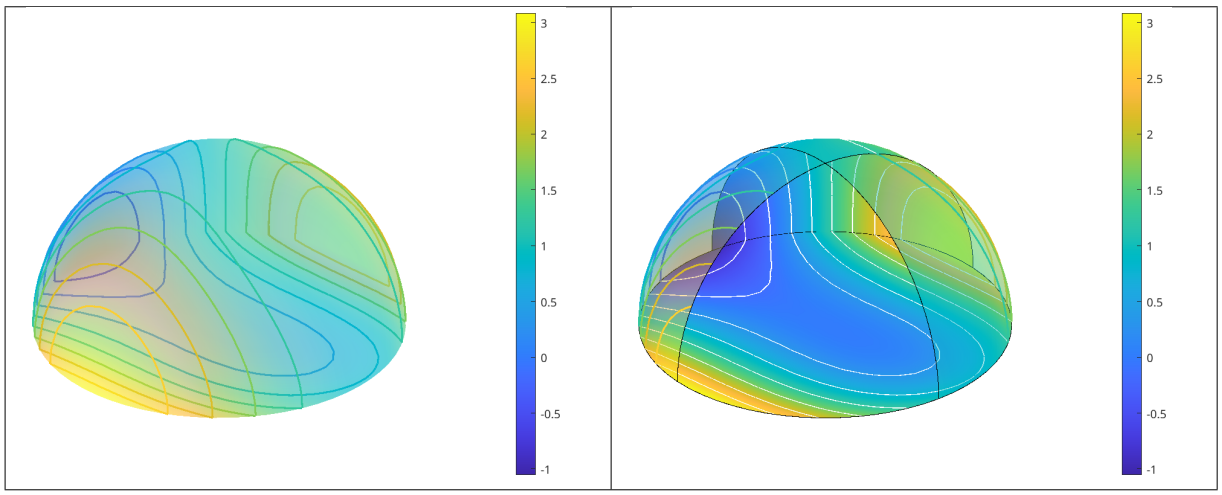
This function accepts until 4 output arguments :

- 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.



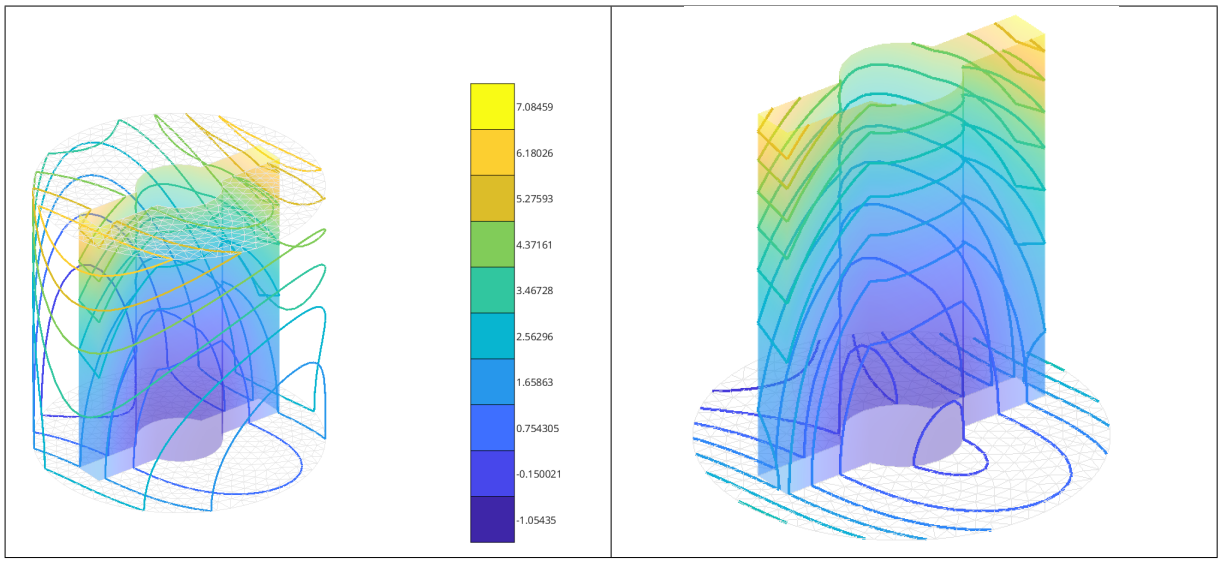

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',25,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=Th.eval(@(x,y) 5*exp(-3*(x.^2+y.^2)).*cos(x).*sin(y));
figure(1)
Th.plotmesh('d',1,'color','k')
hold on;axis off;axis image;
Th.plotiso(u,'isocolorbar',true)
figure(2)
Th.plot(u,'plane',true,'FaceAlpha',0.7)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

Listing 27: Using the **fc_simesh.plotiso** function with a 2D mesh, part of the **fc_simesh.demos.plotiso2D** 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);
figure(1)
Th.plot(u,'FaceColor','interp', 'EdgeColor','none','FaceAlpha',0.4)
hold on;axis off;axis equal;colorbar
Th.plotiso(u,'LineWidth',1.5)
figure(2)
Th.plot(u,'labels',[1,11],'FaceColor','interp', 'EdgeColor','none')
hold on;axis off;axis equal;colorbar;
Th.plotiso(u,'labels',[1,11],'LineWidth',1.,'color','w')
Th.plot(u,'labels',[10,12],'FaceColor','interp', 'EdgeColor','none','FaceAlpha',0.4)
Th.plotiso(u,'labels',[10,12],'LineWidth',1.5)
Th.plotmesh('d',1,'Color','k')
```

Listing 28: Using the fc_simesh.plotiso function with a 3Ds mesh, part of the fc_simesh.demos.plotiso3Ds 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);
figure(1)
Th.plot(u,'d',2,'labels',[10,11,31],'FaceColor','interp', 'EdgeColor','none','FaceAlpha',0.4)
hold on;view(3);axis off;axis equal;
Th.plotmesh('d',2,'labels',[1000,1020,1021,2000,2020,2021], 'FaceColor','none','EdgeColor',0.9*[1,1,1])
Th.plotiso(u,'isocolorbar',true,'LineWidth',1.5)
figure(2)
Th.plot(u,'d',2,'labels',[10,11,31],'FaceColor','interp', 'EdgeColor','none','FaceAlpha',0.4)
hold on;axis off;axis equal;
Th.plotmesh('d',2,'labels',[2000,2020,2021], 'FaceColor','none','EdgeColor',0.9*[1,1,1])
Th.plotiso(u,'labels',[10,11,31,2000,2020,2021],'LineWidth',1.5, 'niso',15)
```

Listing 29: Using the fc_simesh.plotiso function with a 3D mesh, part of the fc_simesh.demos.plotiso3D function

3.4.14 slicemesh method

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

Syntaxe

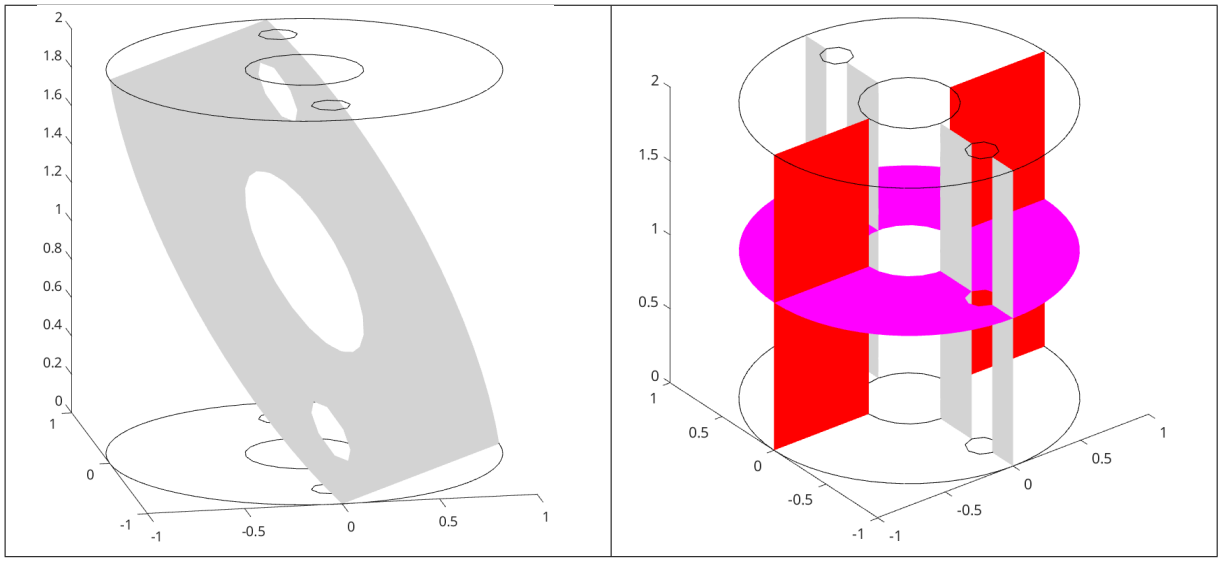
```
Th.slicemesh(P)
Th.slicemesh(P,Name,Value, ...)
```

Description

Th.**slicemesh**(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 toolbox. 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.

Th.**slicemesh**(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,



```
meshfile=fc_oogmsh.gmsh.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]);
Th.slicemesh(P)
hold on;axis equal;
Th.plotmesh('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])];
Th.slicemesh(P,'color',{'LightGray',[1,0,0],'m'}) % [1,0,0] -> RGB => red
hold on;axis equal;
Th.plotmesh('d',1,'color','k')
view(3)
```

Listing 30: Using the fc_simesh.slicemesh function with a 3D mesh, part of the fc_simesh.demos.slicemesh3D function

3.4.15 slice method

The method slice method displays datas on the intersection of a plane and a 3D mesh or parts of a 3D mesh defined by an siMesh object.

Syntaxe

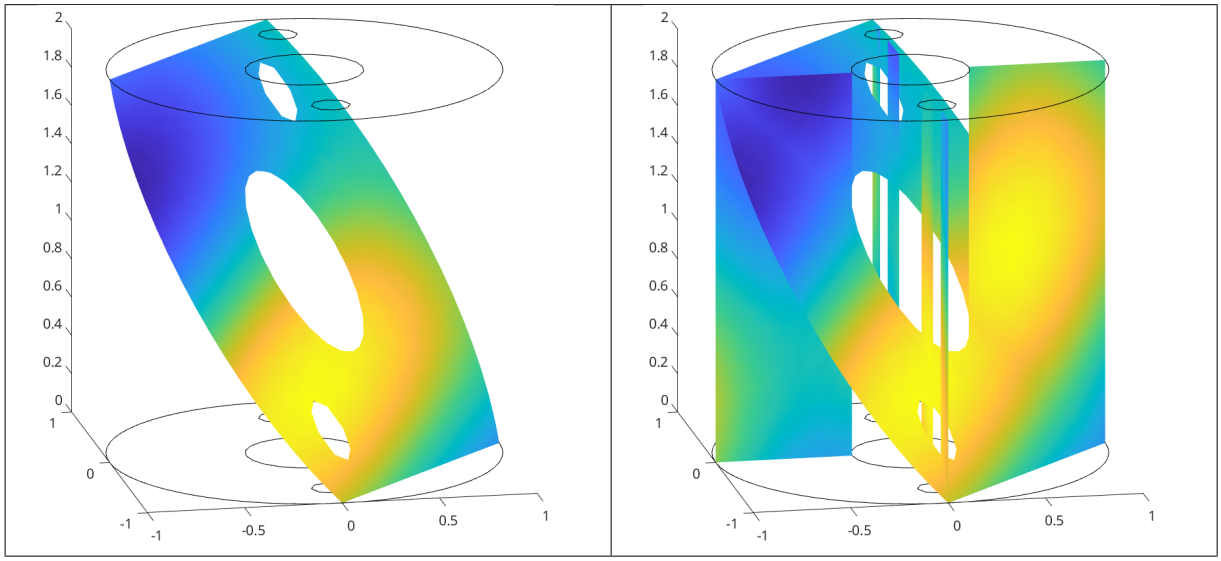
```
Th.slice(u,P)
Th.slice(u,P,Name,Value, ...)
```

Description

Th.**slice**(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 toolbox. 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.

Th.**slice**(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,



```
meshfile=fc_oogmsh.gmsh.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]);
Th.slice(U,P)
hold on
Th.plotmesh('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)
Th.slice(U,P)
hold on
Th.plotmesh('d',1,'color','k')
axis equal
view(-11,15)
```

Listing 31: Using the fc_simesh.slice function with a 3D mesh, part of the fc_simesh.demos.slice3D function

3.4.16 sliceiso method

The sliceiso method 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

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

Description

Th.**sliceiso**(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 toolbox. 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 the plane.

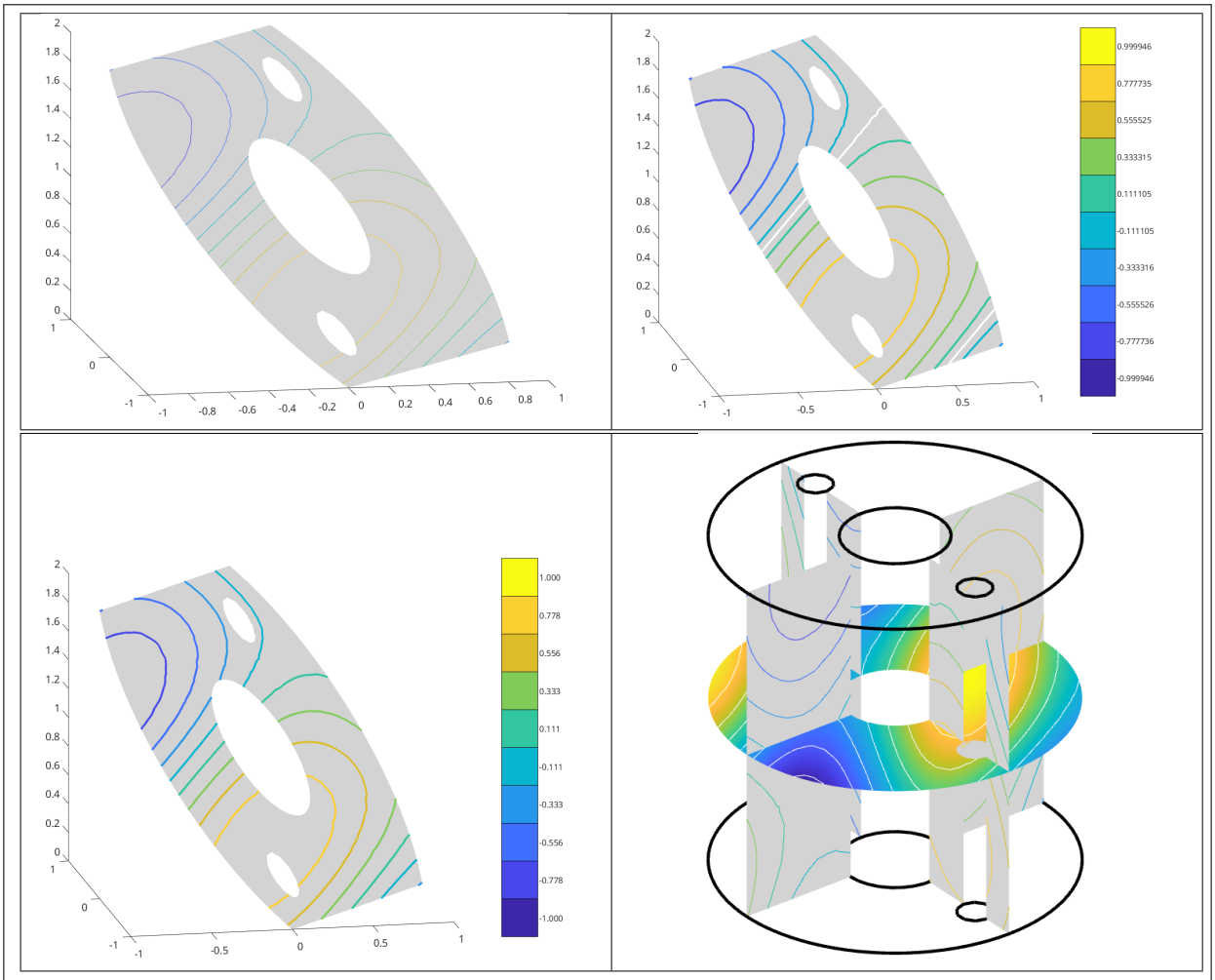
Th.**sliceiso**(u,P,key,value, ...) allows additional key/value pairs to be used when displaying u. The key strings could be

- '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.



```
meshfile=fc_oogmsh.gmsh.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(2,2,'figures',1:4);
figure(1)
P=fc_tools.graphics.PlaneCoefs([0 0 1],[1 -1 1]);
Th.slicemesh(P)
hold on
Th.sliceiso(U,P)
view(-11,15)

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

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])];

figure(4)
Th.plotmesh('d',1,'LineWidth',2,'color','k');
hold on;axis off;axis image;
Th.slicemesh(P(1:2,:))
Th.sliceiso(u,P(1:2,:))
Th.slice(u,P(3,:),'Facecolor','interp')
```

Listing 32: Using the fc_simesh.sliceiso function with a 3D mesh, part of the fc_simesh.demos.sliceiso3D function

3.4.17 plotquiver method

The **plotquiver** method displays vector field datas on the mesh or parts of the mesh defined by an siMesh object.

Syntaxe

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

Description

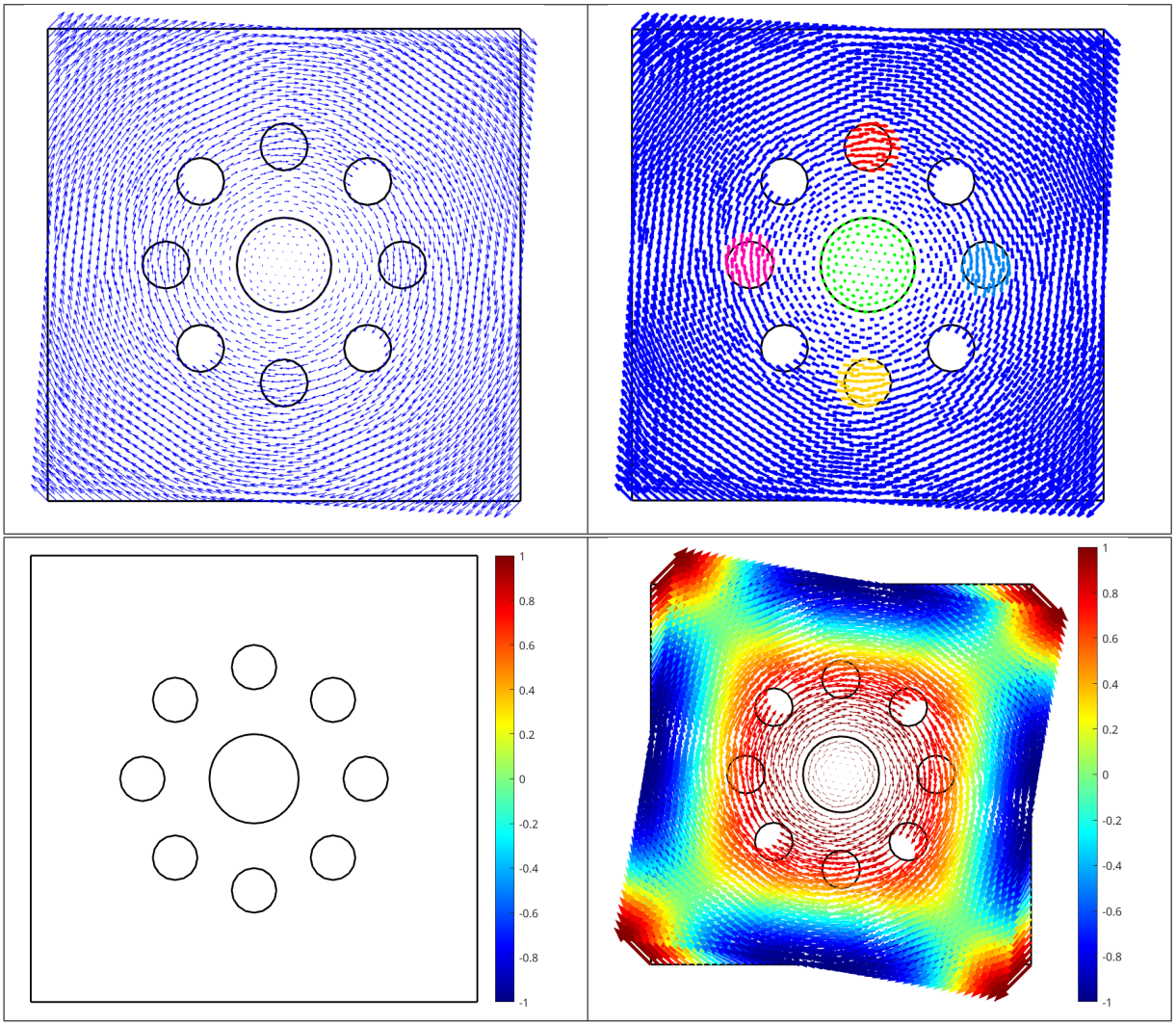
Th.**plotquiver**(V) displays vector field U on all the d-dimensional simplices elements in dimension $d = 2$ or $d = 3$. The data V is an 2D-array of size Th.nq-by-d or 2-by-Th.nq.

Th.**plotquiver**(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).

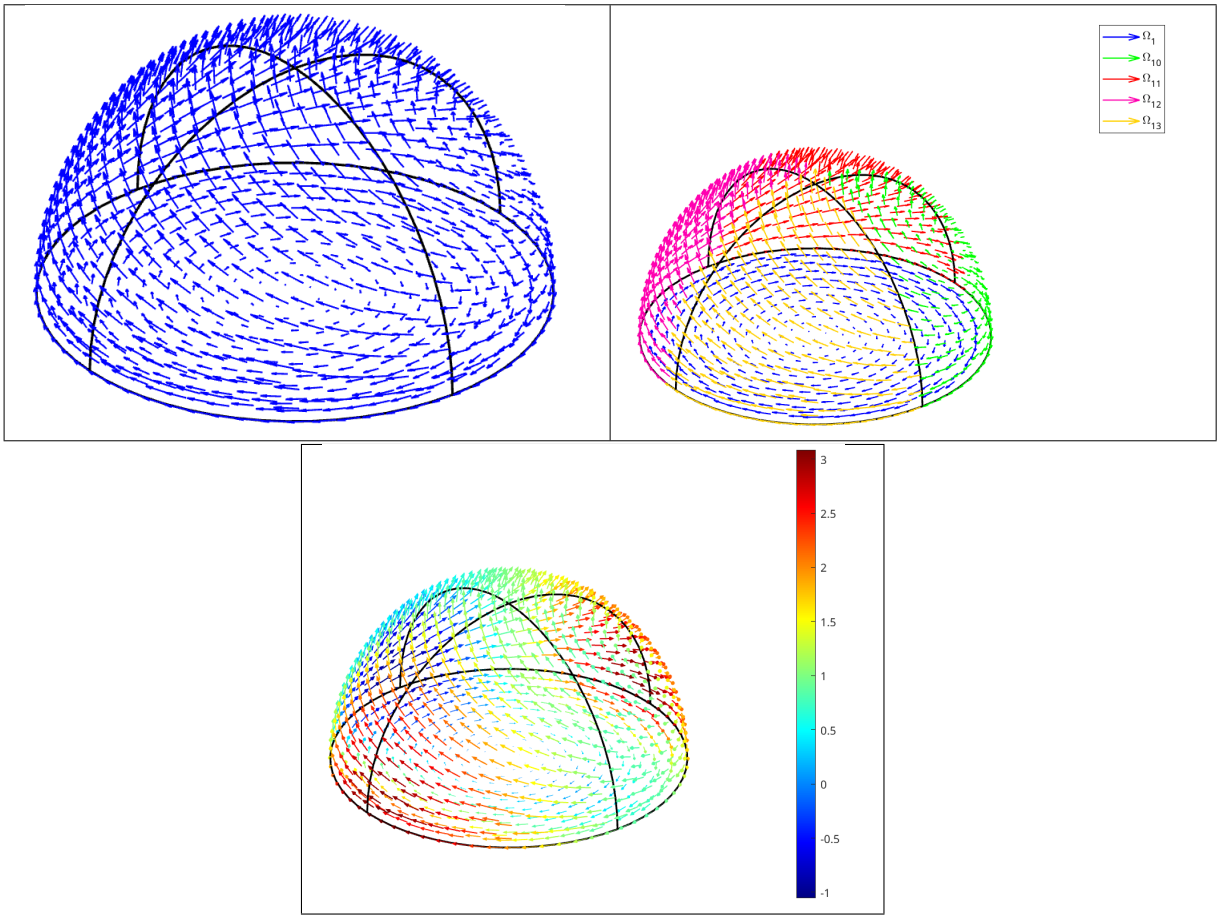
The options of second level depend on space dimension and 'colordata' option. One can use any option of the following functions

- **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.



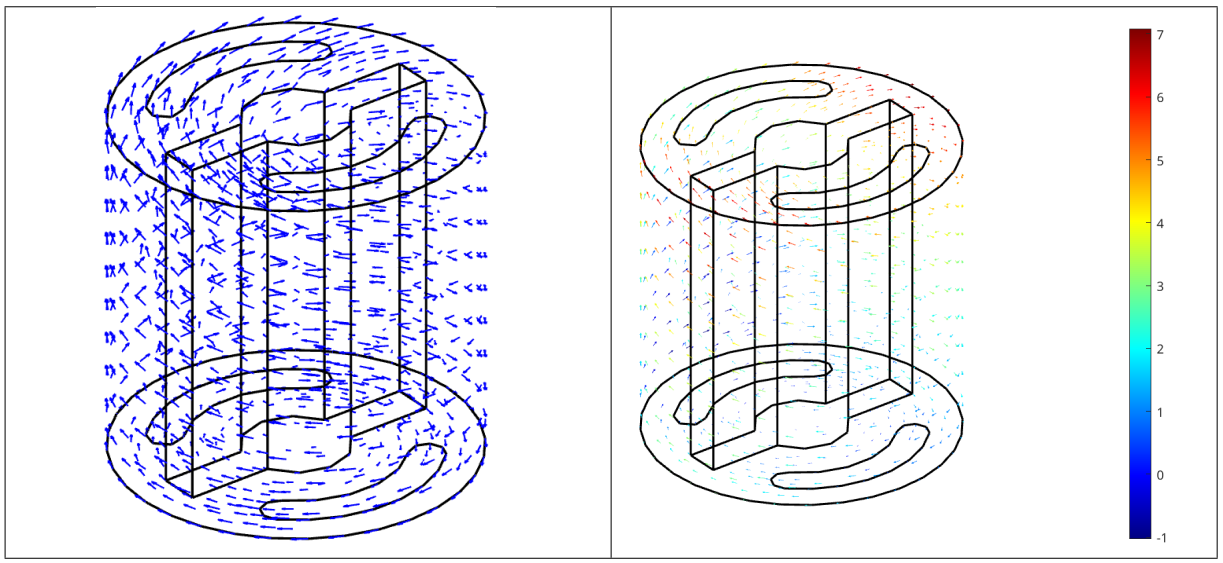

```
geofile=fc_simesh.get_geo(2,2,'condenser11');
meshfile=fc_oogmsh.buildmesh2d(geofile,25,varargin{:});
Th=fc_simesh.simMesh(meshfile);

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);

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

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

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

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

Listing 33: Using the **fc_simesh.plotquiver** function with a 2D mesh, part of the **fc_simesh.demos.plotquiver2D** function




```
geofile=fc_simesh.get_geo(3,2,'demisphere5');
meshfile=fc_oogmsh.gmsh.buildmesh3ds(geofile,10,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 };
W=Th.eval(w);
u=Th.eval(@(x,y,z) 3*x.^2-y.^3+z.^2+x.*y);

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

figure(2)
Th.plotmesh('d',1,'color','k', 'LineWidth',1.5)
hold on;axis off;axis image
Th.plotquiver(W,'LineWidth',1,'merge',false,'inlegend',true)
fc_graphics4mesh.legend()

figure(3)
Th.plotmesh('d',1,'color','k', 'LineWidth',1.5)
hold on;axis off;axis image
Th.plotquiver(W,'colordata',u, 'scale',0.1)
colormap('jet');colorbar
```

Listing 34: Using the fc_simesh.plotquiver function with a 3Ds mesh, part of the fc_simesh.demos.plotquiver3Ds function




```
geofile=fc_simesh.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)
Th.plotmesh('d',1,'color','k', 'LineWidth',1.5)
hold on
Th.plotquiver(W,'LineWidth',1)
axis off;axis image

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

Listing 35: Using the fc_simesh.plotquiver function with a 3D mesh, part of the fc_simesh.demos.plotquiver3D function

3.4.18 scatter method

The scatter method displays scalar datas as colorized points on the mesh or parts of the mesh defined by an siMesh object.

Syntaxe

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

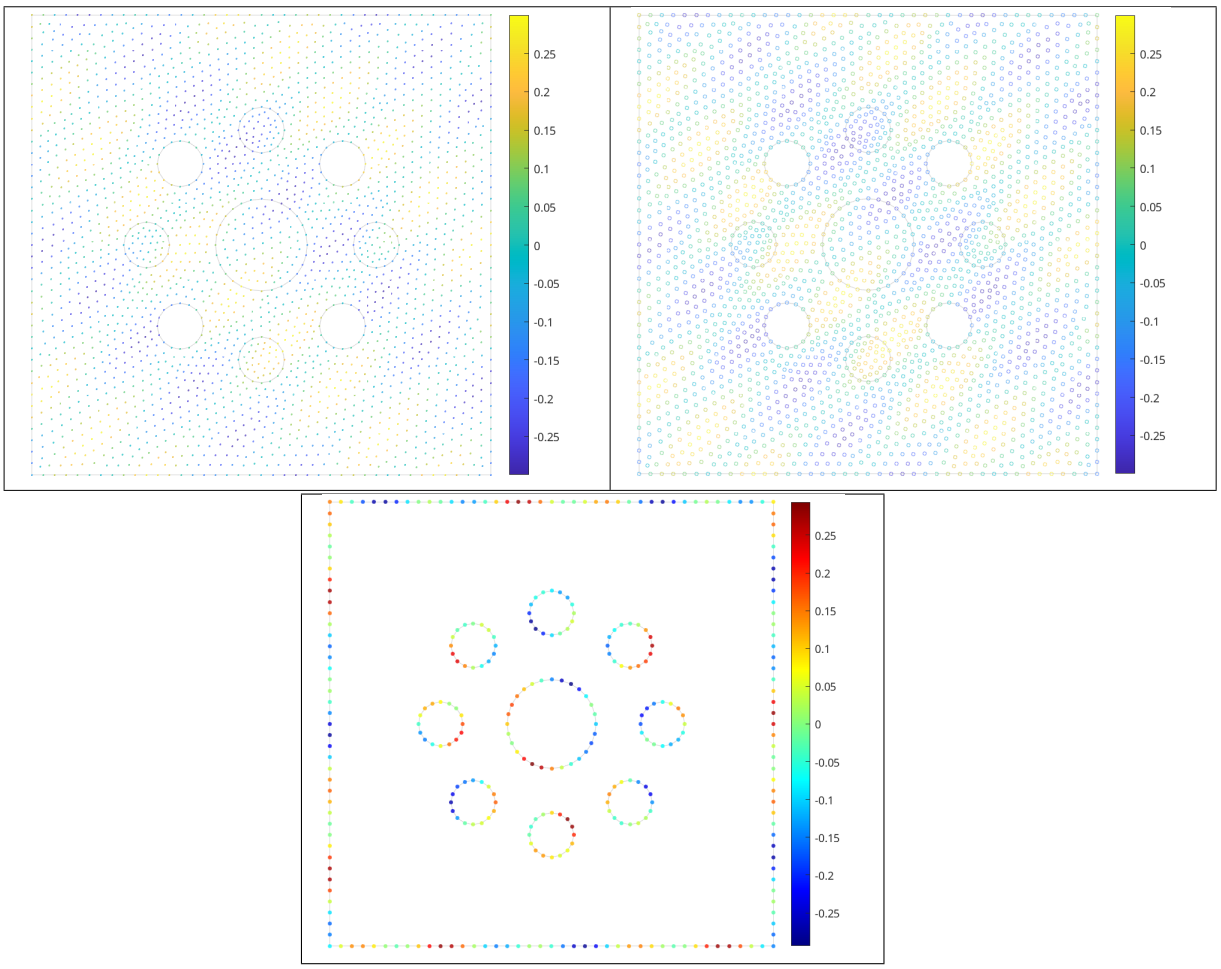
Description

Th.scatter(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.

Th.scatter(u,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options of first level are

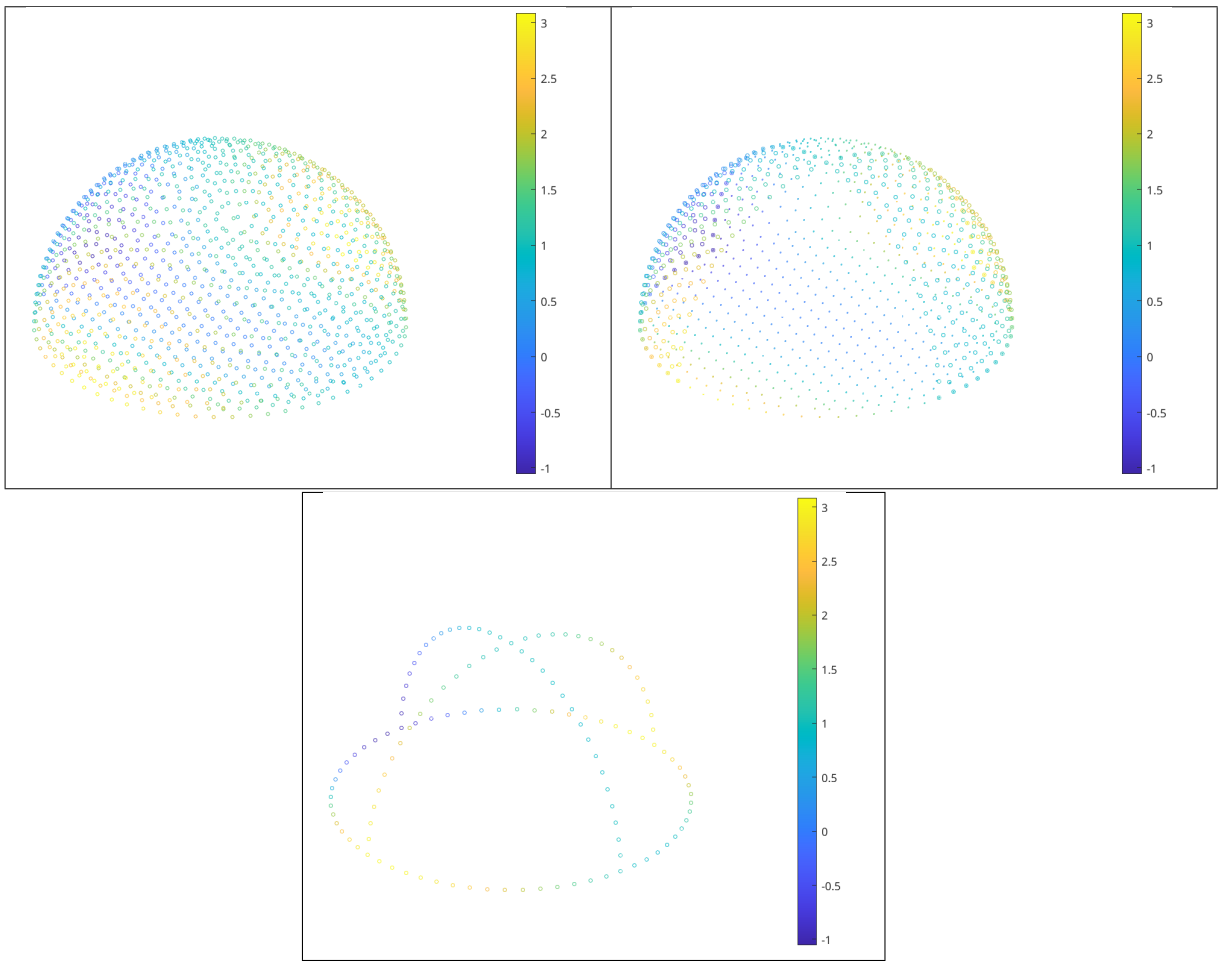
- 'd' : to specify the dimension of the simplices elements (default : Th.d)
- 'labels' : to select the labels of the elements to display data,
- 'MarkerSize' : size of the marker. Default is 1.
- 'ForcePatch' : if **true**, uses **patch** function, otherwise uses scatter function in dimension 2 or scatter3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

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



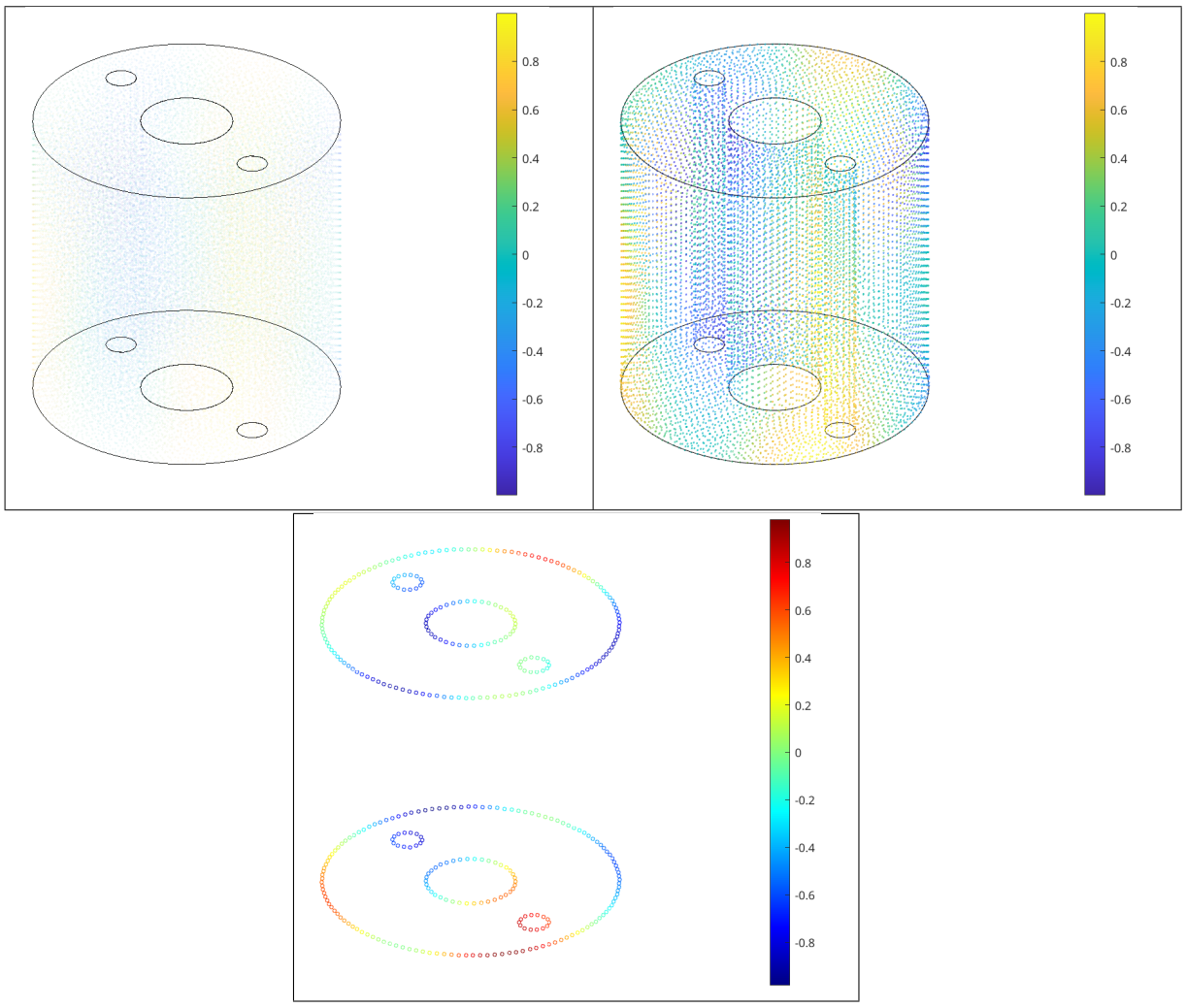

```
meshfile=fc_oogmsh.gmsh.buildmesh2d('condenser11',20,varargin{:});
Th=fc_simesh.siMesh(meshfile);
u=@(x,y) 0.3*cos(5*(2*x-y)).*sin(5*(x-2*y));
U=Th.eval(u);
figure(1)
Th.plotmesh('d',1,'color','LightGray')
hold on
Th.scatter(U);
axis image;axis off;
colorbar
figure(2)
Th.plotmesh('d',1,'color','LightGray')
hold on  % '*Alpha' options not yet implemented under Octave
Th.scatter(U,'MarkerSize',8,'MarkerEdgeAlpha',0.6);
axis off;axis image
colorbar
figure(3)
Th.plotmesh('d',1,'color','LightGray')
hold on  % '*Alpha' options not yet implemented under Octave
Th.scatter(U,'d',1,'MarkerSize',12,'MarkerEdgeAlpha',0,'MarkerFaceColor', 'flat','MarkerFaceAlpha',0.8)
axis off;axis image
colormap('jet')
colorbar
```

Listing 36: Using the **fc_simesh.scatter** function with a 2D mesh, part of the **fc_simesh.demos.scatter2D** 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);
figure(1)
Th.scatter(u,'MarkerSize',7)
view(3);axis off;axis image;colorbar;
figure(2)
Th.scatter(u,'labels',[1,11])
hold on;axis off;axis image;colorbar;
Th.scatter(u,'labels',[10,12], 'MarkerSize',9)
view(3)
figure(3)
Th.scatter(u,'d',1,'MarkerSize',9)
axis off;axis image;colorbar
view(3)
```

Listing 37: Using the fc_simesh.scatter function with a 3Ds mesh, part of the fc_simesh.demos.scatter3Ds 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);
figure(1)
Th.plotmesh('d',1,'color','k')
hold on
Th.scatter(U,'MarkerEdgeAlpha',0.1);
axis image;axis off;view(3)
colorbar
figure(2)
Th.plotmesh('d',1,'color','k')
hold on
Th.scatter(U,'d',2);
axis image;axis off;view(3)
colorbar
figure(3)
colormap('jet')
Th.scatter(U,'d',1,'MarkerSize',8);
hold on
Th.scatter(U,'d',2,'MarkerSize',5,'labels',[10,20,21]);
axis image;axis off;view(3)
colorbar
```

Listing 38: Using the fc_simesh.scatter function with a 3D mesh, part of the fc_simesh.demos.scatter3D function

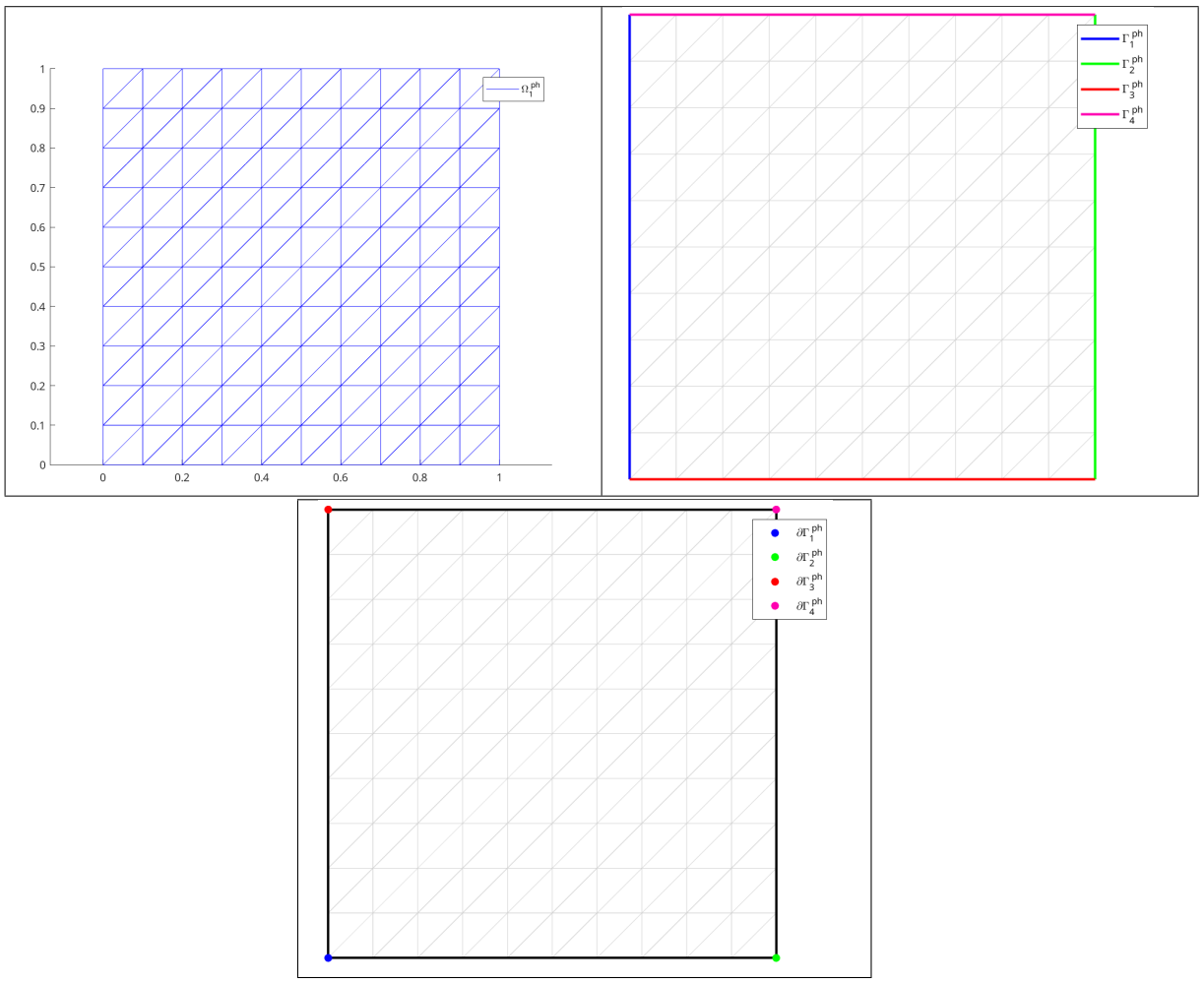
3.5 Hypercube as a fc_simesh.siMesh object

The function fc_simesh.constructor.hypercube allows to create a siMesh object representing an hypercube in any dimension. It uses the fc-hypermesh Matlab toolbox.

- Th=fc_simesh.constructor.hypercube(dim,N) : return a siMesh object representing an hypercube in dimension dim and ...
- Th=fc_simesh.constructor.hypercube(dim,N,Key,Value,...) :

3.5.1 2D hypercube

In Listing 1 a usage example generating a 2D hypercube as a siMesh object is given with reprensentation of its elementary meshes.

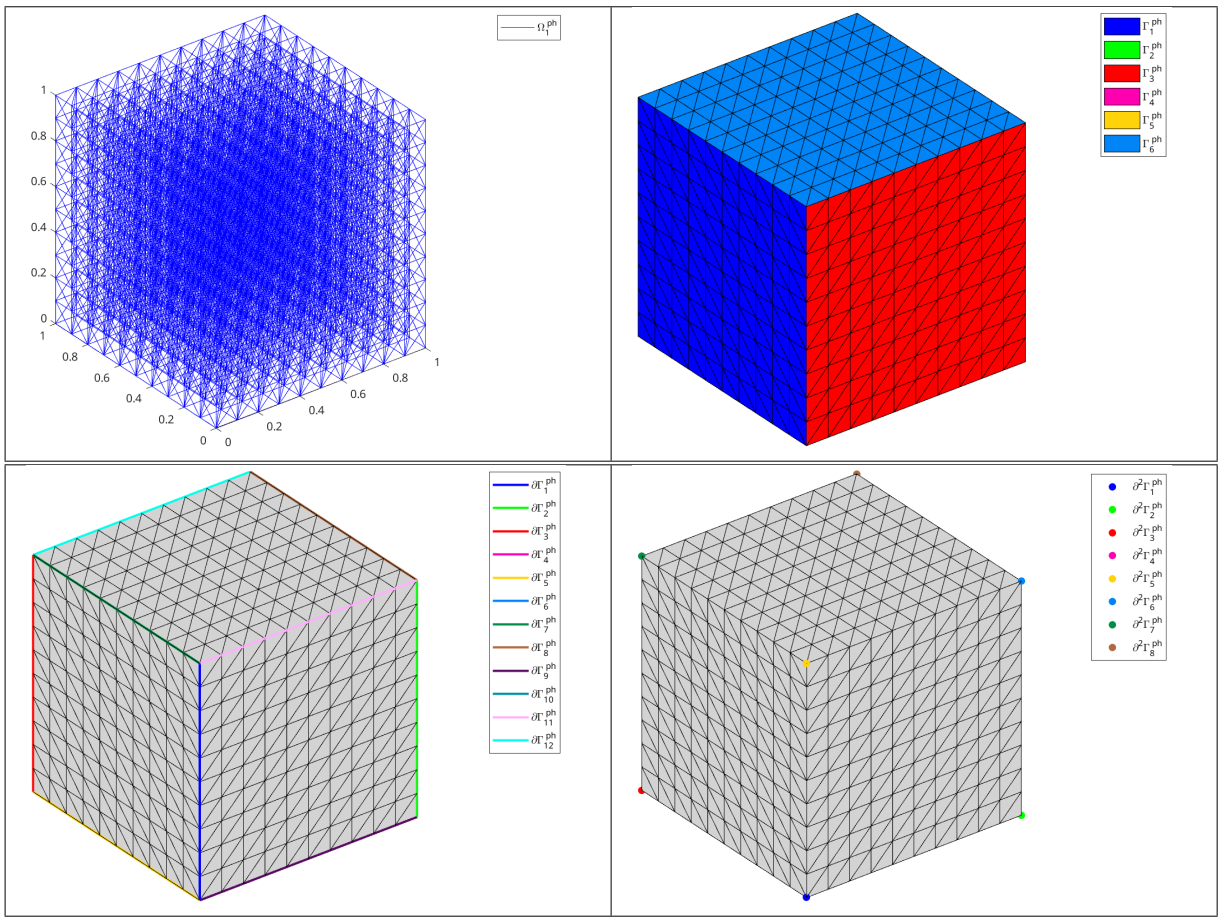



```
Th=fc_simesh.constructor.hypercube(2,10);
figure(1)
Th.plotmesh('inlegend',true)
axis equal;fc_graphics4mesh.legend()
figure(2)
Th.plotmesh('color','LightGray')
hold on;axis equal;axis off;
Th.plotmesh('d',1,'linewidth',2,'inlegend',true)
fc_graphics4mesh.legend()
figure(3)
Th.plotmesh('color','LightGray')
hold on;axis equal;axis off;
Th.plotmesh('d',1,'color','Black','linewidth',2)
Th.plotmesh('d',0,'inlegend',true)
fc_graphics4mesh.legend()
```

Listing 39: 2D Hypercube siMesh object returned by the function **fc_simesh**.constructor.**hypercube** with representation of the elementary meshes with 2-simplices (top left), 1-simplices (top right) and 0-simplices (bottom) (code is part of the **fc_simesh.demos.hypercube2D** function)

3.5.2 3D hypercube

In Listing 1 a usage example generating a 3D hypercube as a siMesh object is given with reprensentation of its elementary meshes.



```
Th=fc_simesh.constructor.hypercube(3,10);
figure(1)
Th.plotmesh('inlegend',true)
axis equal;fc_graphics4mesh.legend()
figure(2)
Th.plotmesh('d',2,'inlegend',true)
hold on;axis equal;axis off;
fc_graphics4mesh.legend()
figure(3)
Th.plotmesh('d',2,'color','LightGray')
hold on;axis equal;axis off;
Th.plotmesh('d',1,'linewidth',2,'inlegend',true)
fc_graphics4mesh.legend()
figure(4)
Th.plotmesh('d',2,'color','LightGray')
hold on;axis equal;axis off;
Th.plotmesh('d',0,'inlegend',true)
fc_graphics4mesh.legend()
```

Listing 40: 3D Hypercube siMesh object returned by the function **fc_simesh**.constructor.**hypercube** with representation of the elementary meshes with 3-simplices (top left), 2-simplices (top right), 1-simplices (bottom left) and 0-simplices (bottom right) (code is part of the **fc_simesh.demos.hypercube3D** function)

3.5.3 4D hypercube

In Listing 41 a usage example generating a 4D hypercube as a siMesh object is given.

Listing 41: : function **fc_simesh**.constructor.hypercube

```
Th=fc_simesh.constructor.hypercube(4,10)   ;
disp(Th)
```

Output

```
fc_simesh.siMesh with properties:
            d: 4 double
          dim: 4 double
           sTh: (1x81 cell)
         nsTh: 81 double
     toGlobal: (1x14641 double)
     toParent: (1x14641 double)
      sThsimp: (1x81 double)
       sThlab: (1x81 double)
    sThcolors: (81x3 double)
         bbox: [ 0 1 0 1 0 1 0 1 ] (1x8 double)
    sThgeolab: (1x81 double)
   sThphyslab: (1x81 double)
   sThpartlabs: []
  sThboundlabs: []
           nq: 14641 double
    nqParents: 14641 double
    toParents: (1x1 cell)
        other: []
```

3.5.4 5D hypercube

In Listing 42 a usage example generating a 5D hypercube as a siMesh object is given.

Listing 42: : function **siMesh**.constructor.hypercube

```
Th=fc_simesh.constructor.hypercube(5,6);
disp(Th)
```

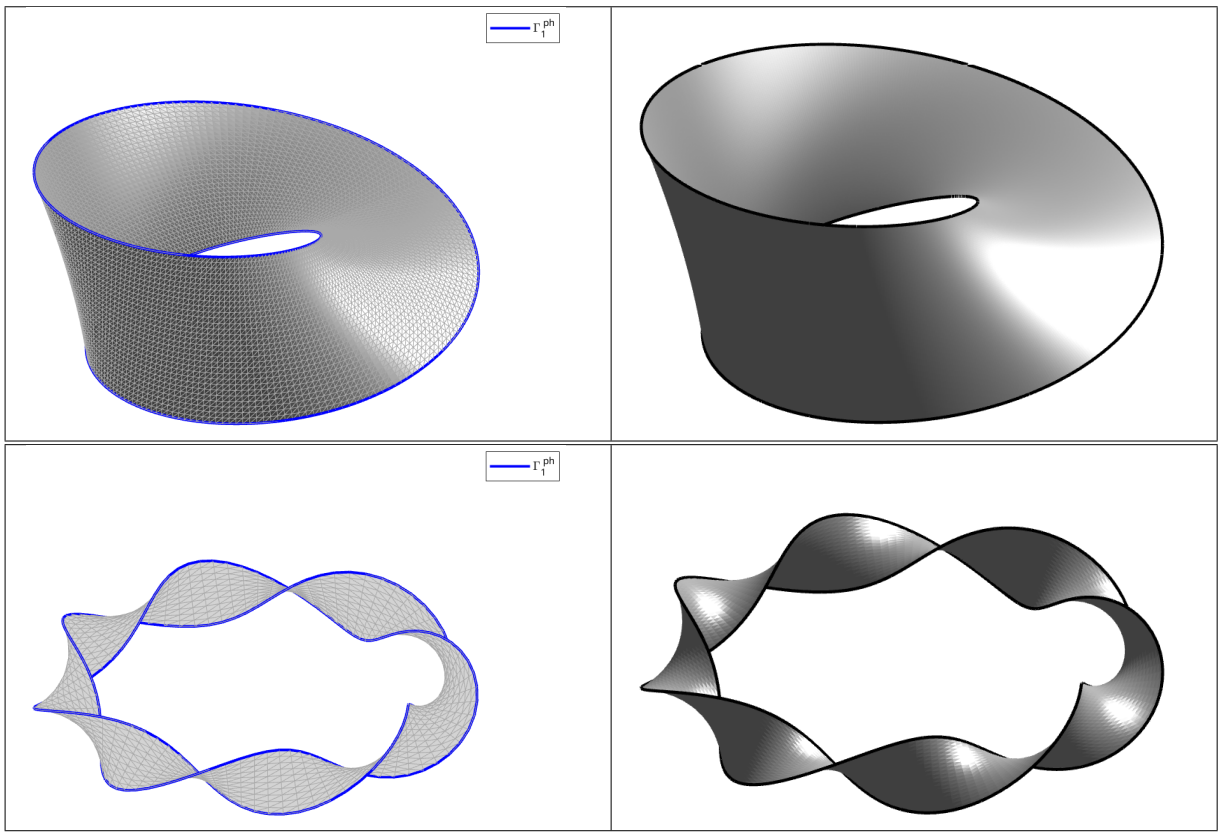
Output

```
fc_simesh.siMesh with properties:
            d: 5 double
          dim: 5 double
           sTh: (1x243 cell)
         nsTh: 243 double
     toGlobal: (1x16807 double)
     toParent: (1x16807 double)
      sThsimp: (1x243 double)
       sThlab: (1x243 double)
    sThcolors: (243x3 double)
         bbox: [ 0 1 0 1 0 1 0 1 0 1 ] (1x10 double)
    sThgeolab: (1x243 double)
   sThphyslab: (1x243 double)
   sThpartlabs: []
  sThboundlabs: []
           nq: 16807 double
    nqParents: 16807 double
    toParents: (1x1 cell)
        other: []
```

3.5.5 Möbius strip as a fc_simesh.siMesh object

The function **fc_simesh**.constructor.mobius allows to create a siMesh object representing a Möbius strip in dimension 3.

- Th=**fc_simesh**.constructor.mobius(N) : return a siMesh object representing a mobious strip. The N value is the number of discretisations in θ (angle) and r (radius). if N is an array of length 2, then N(1) and N(2) are respectively the number of discretisations in θ and the number of discretisations in r. If N is a scalar then N is taken for the both values.
- Th=**fc_simesh**.constructor.mobius(N,Name,Value,...) : specifies function options using one or more Name,Value pair arguments,
  - 'radius' : to specify the radius of the mid circle. Default is 1.
  - 'width' : to specify the width of the strip.




```
Th=fc_simesh.constructor.mobius(30); % :start:
figure(1)
Th.plotmesh('FaceColor','LightGray','EdgeColor','DarkGray')
hold on
Th.plotmesh('d',1,'inlegend',true,'LineWidth',2)
axis equal;axis off;
fc_graphics4mesh.legend('Location','northeast')
light

Th=fc_simesh.constructor.mobius(100);
figure(2)
Th.plotmesh('FaceColor','LightGray','EdgeColor','None')
hold on
Th.plotmesh('d',1,'color','k','LineWidth',2)
axis equal;axis off;
light

Th=fc_simesh.constructor.mobius([10,90],'k',3,'radius',3,'width',1);
figure(3)
Th.plotmesh('FaceColor','LightGray','EdgeColor','DarkGray')
hold on
Th.plotmesh('d',1,'inlegend',true,'LineWidth',2)
axis equal;axis off;
fc_graphics4mesh.legend('Location','northeast')

Th=fc_simesh.constructor.mobius([20,180],'k',3,'radius',3,'width',1);
figure(4)
Th.plotmesh('FaceColor','LightGray','EdgeColor','None')
hold on
Th.plotmesh('d',1,'color','k','LineWidth',2)
axis equal;axis off;
light
```

Listing 43: Using the fc_simesh.mobius function with a 3D mesh, part of the fc_simesh.demos.mobius3D function

4 Other meshes

4.1 2-simplicial meshes in $\mathbb{R}^2$

4.1.1 fc_simesh.samples.square function

The fc_simesh.samples.square function returns an siMesh object corresponding to a square obtain by using gmsh with rectangleOC.geo file.

Syntaxe

```
Th=fc_simesh.samples.square(N)
Th=fc_simesh.samples.square(N,Name,Value, ...)
```

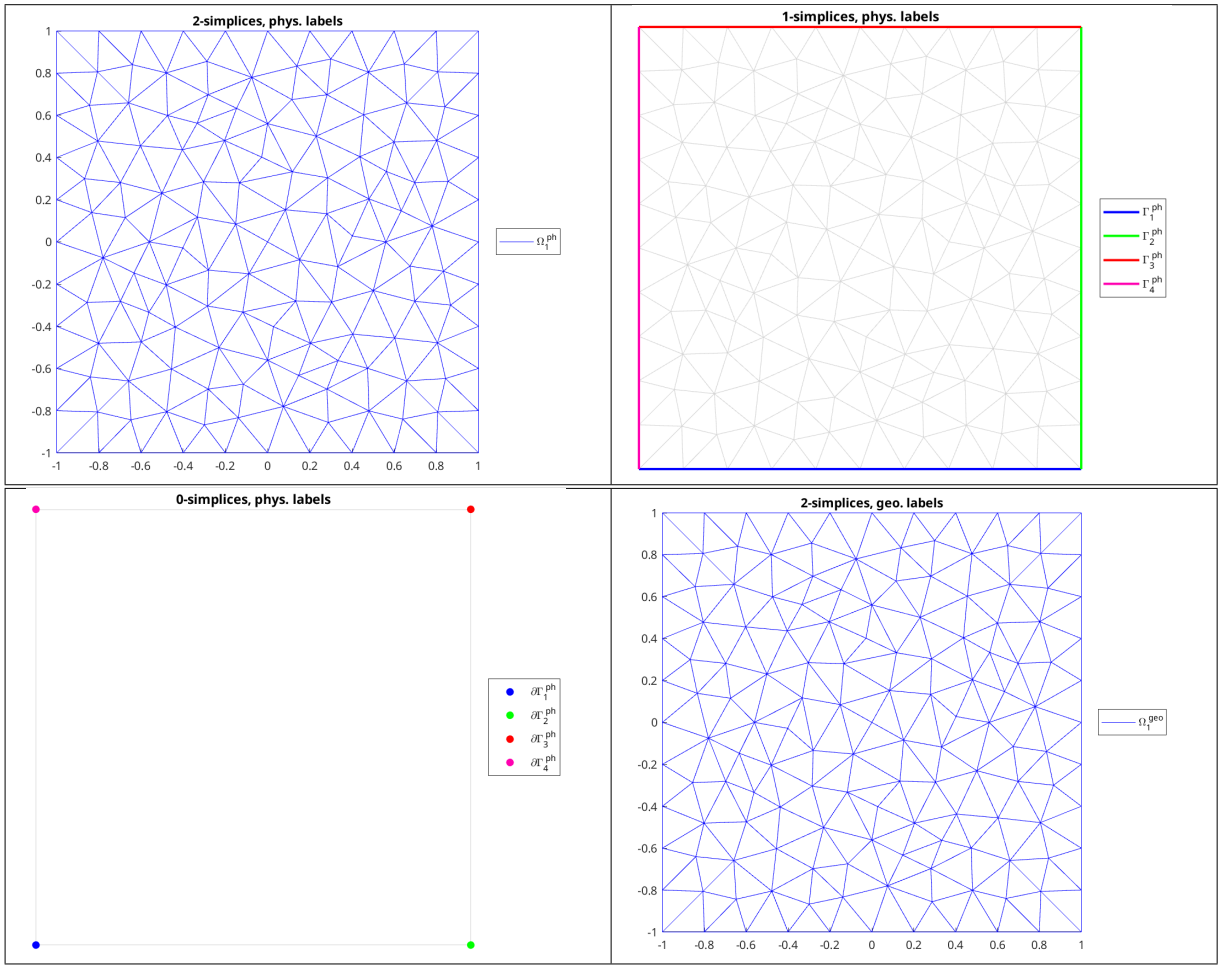
Description

fc_simesh.samples.square(N) returns the unit square, $[0,1]\times[0,1]$, as a siMesh object where N is a refinement parameter.

fc_simesh.samples.square(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'L' : to specify the side length of the square (default 1),
- 'x' : to specify the x-value of the bottom-left corner (default 0),
- 'y' : to specify the y-value of the bottom-left corner (default 0),
- 'verbose' : to specify verbosity from 0 to 4... (default 0)
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Figure 44 for an example.

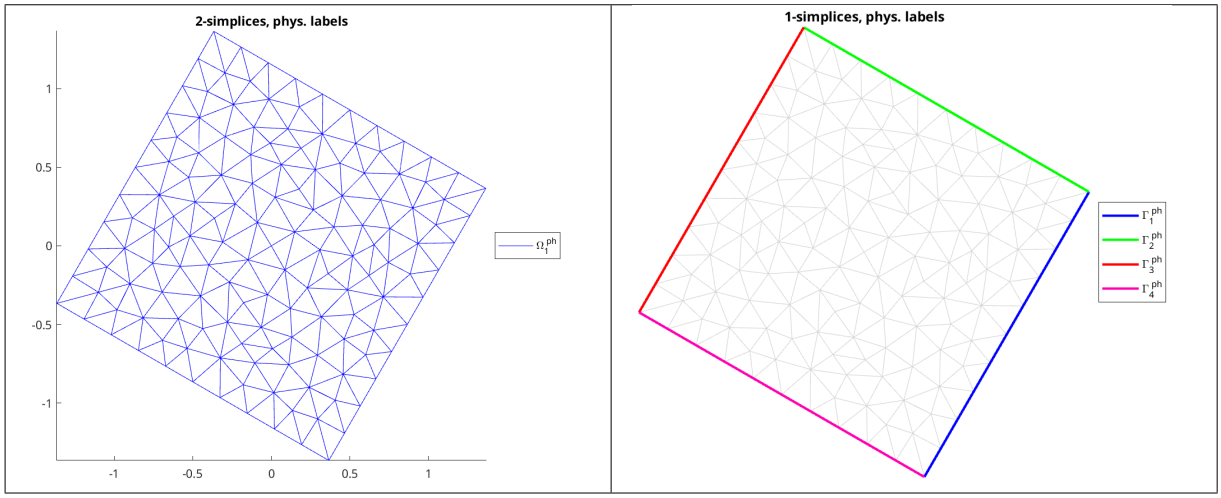



```
Th=fc_simesh.samples.square(20,'L',2,'x',-1,'y',-1);
Th.plotmeshinfo()
```

Listing 44: siMesh object corresponding to the square $[-1,1]\times[-1,1]$, geometrical labels (top) and boundary physical labels (bottom).

remark 4.1

To rotate by θ, one can use the transform method.




```
Th=fc_simesh.samples.square(20,'L',2,'x',-1,'y',-1);
Rot=@(t) [cos(t),-sin(t);sin(t),cos(t)]; % Rotation matrix
Th.transform(@(q) Rot(pi/3)*q);
Th.plotmeshinfo()
```

Listing 45: siMesh object corresponding to the square $[-1,1] \times [-1,1]$ rotated by $\pi/3$.

4.1.2 fc_simesh.samples.rectangle function

The **fc_simesh**.samples.rectangle function returns an siMesh object corresponding to a rectangle obtain by using `gmsh` with `rectangleOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.rectangle(N)
Th=fc_simesh.samples.rectangle(N,Name,Value, ...)
```

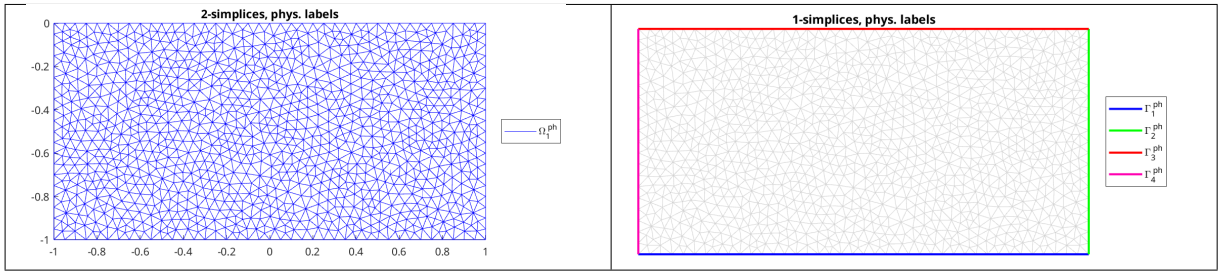
Description

fc_simesh.samples.rectangle(N) returns the rectangle, $[x, x + L_x] \times [y, y +_L y]$, as a siMesh object where N is a refinement parameter.

fc_simesh.samples.rectangle(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'Lx' : to specify the side length along x-axis of the rectangle (default 1),
- 'Ly' : to specify the side length along y-axis of the rectangle (default 1),
- 'x' : to specify the x-value of the bottom-left corner (default 0),
- 'y' : to specify the y-value of the bottom-left corner (default 0),
- 'verbose' : to specify verbosity from 0 to 4... (default 0)
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Figure 46 for an example.




```
Th=fc_simesh.samples.rectangle(40,'Lx',2,'Ly',1,'x',-1,'y',-1);
Th.plotmeshinfo()
```

Listing 46: siMesh object corresponding to the rectangle $[-1,1] \times [-1,1]$.

remark 4.2

To rotate by θ, one can use the transform method (see Listing 44).

4.1.3 fc_simesh.samples.disk function

The **fc_simesh**.samples.disk function returns an siMesh object corresponding to a disk obtain by using `gmsh` with `diskOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.disk(N)
Th=fc_simesh.samples.disk(N,Name,Value, ...)
```

Description

fc_simesh.samples.disk(N) returns the unit disk, $\mathcal{C}([0,0],1)$, as a siMesh object where N is a refinement parameter.

fc_simesh.samples.disk(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

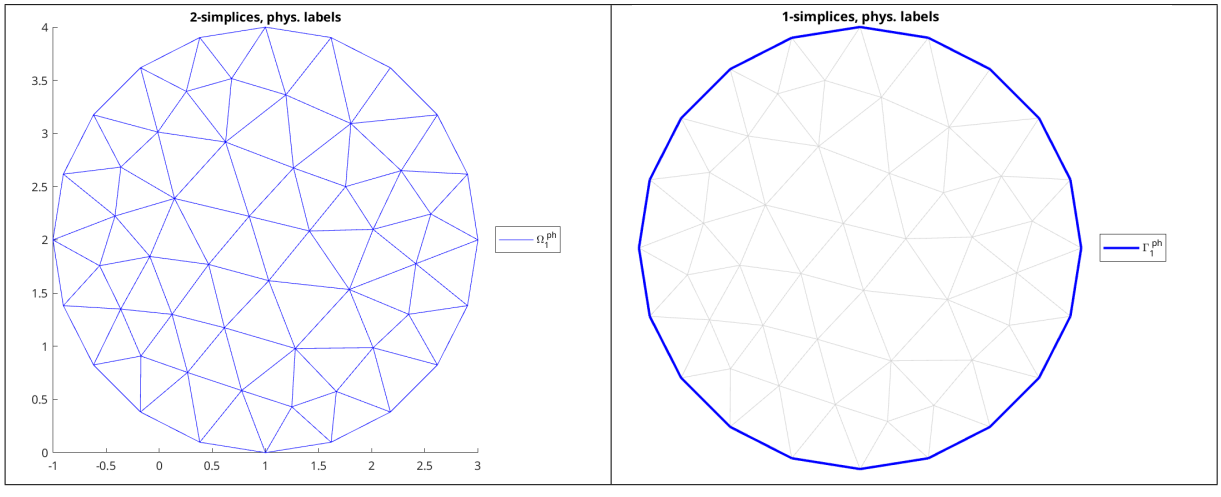
- 'R' : to specify the radius of the disk $\mathcal{C}([0,0],R)$,
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Figure 47 for an example.

remark 4.3

To change center of the disk to $C = (\mathrm{cx};\mathrm{cx})$, one can use the transform method

Th.transform(@(q) q+[cx;cy]);




```
Th=fc_simesh.samples.disk(20,'R',2);
Th.transform(@(q) q+[1;2]);
Th.plotmeshinfo()
```

Listing 47: siMesh object corresponding to the disk of center $(1,2)$ and radius 2.

4.1.4 fc_simesh.samples.ellipse function

The **fc_simesh**.samples.ellipse function returns an siMesh object corresponding to a ellipse obtain by using `gmsh` with `ellipseOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.ellipse(N)
Th=fc_simesh.samples.ellipse(N,Name,Value, ...)
```

Description

fc_simesh.samples.ellipse(N) returns an ellipse, centered in $(0,0)$ with $R_x = 1$ radius along x-axis ans $R_y = 0.75$ radius along y-axis, as a siMesh object where N is a refinement parameter.

fc_simesh.samples.ellipse(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

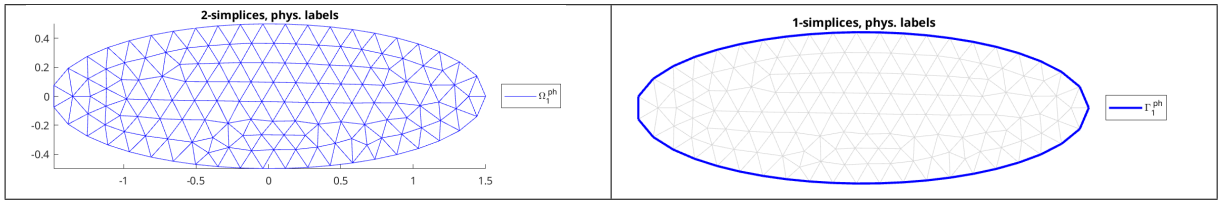
- 'Rx' : to specify the radius along x-axis (default 1),
- 'Ry' : to specify the radius along y-axis (default 0.75),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Listings 48 and 48..

remark 4.4

To change center of the ellipse to $C = (cx; cx)$, one can use the transform method

Th.transform(@(q) q+[cx;cy]);

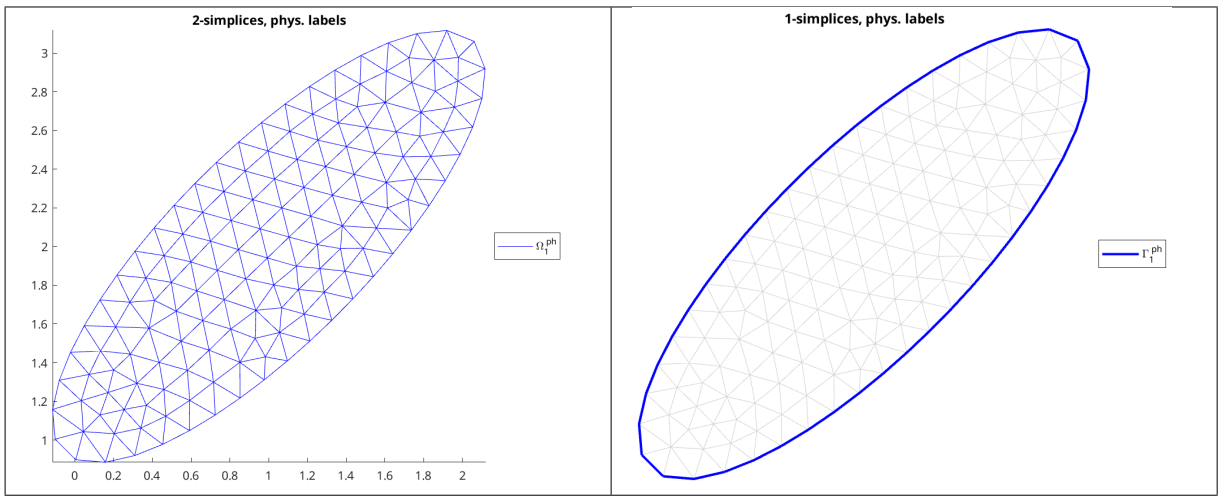



```
Th=fc_simesh.samples.ellipse(20,'Rx',1.5,'Ry',0.5);
Th.plotmeshinfo()
```

Listing 48: siMesh object corresponding to an ellipse centered on $(0,0)$, with radius 1.5 along x-axis and radius 0.5 along y-axis.

remark 4.5

To rotate by θ and to change center of the ellipse to $C = (cx; cx)$, one can use the transform method.




```
Th=fc_simesh.samples.ellipse(20,'Rx',1.5,'Ry',0.5);
Rot=@(t) [cos(t),-sin(t);sin(t),cos(t)]; % Rotation matrix
Th.transform(@(q) Rot(pi/4)*q+[1;2]);
Th.plotmeshinfo()
```

Listing 49: siMesh object corresponding to an ellipse centered on $(0,0)$, with radius 1.5 along x-axis and radius 0.5 along y-axis, rotated by $\pi/4$ and move by $(1,2)$ vector.

4.1.5 fc_simesh.samples.ring function

The **fc_simesh**.samples.ring function returns an siMesh object corresponding to a ring obtain by using `gmsh` with `ringOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.ring(N)
Th=fc_simesh.samples.ring(N,Name,Value, ...)
```

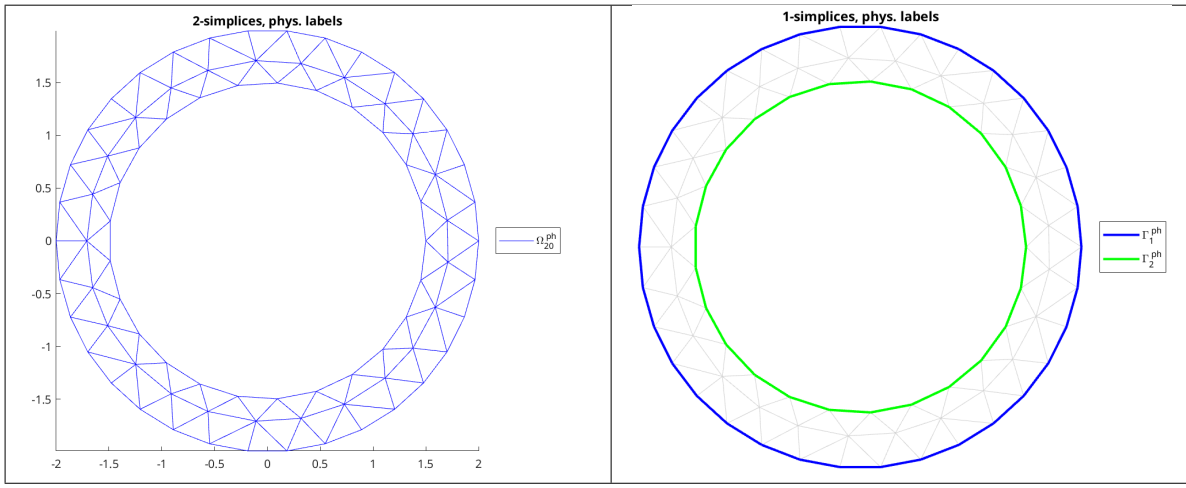
Description

fc_simesh.samples.ring(N) returns a ring, centered in $[0,0]$ with outer radius $R_o = 1$ and inner radius $R_i = 0.4$ as a siMesh object where N is a refinement parameter.

fc_simesh.samples.ring(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'Ro' : to specify the outer radius (default 1)
- 'Ri' : to specify the inner radius (default 0.4)
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Figure 50 for an example.




```
Th=fc_simesh.samples.ring(50,'Ro',2,'Ri',1.5);
Th.plotmeshinfo()
```

Listing 50: siMesh object corresponding to the ring with outer radius $R_o = 2$ and inner radius $R_i = 1.5$.

remark 4.6

To change center of the ring to $C = (cx; cx)$, one can use the transform method

```
Th.transform(@(q) q+[cx;cy]);
```

4.1.6 fc_simesh.samples.regular_polygon function

The **fc_simesh**.samples.regular_polygon function returns an siMesh object corresponding to a regular polygon obtain by using `gmsh` with `regular_polygonOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.regular_polygon(N)
Th=fc_simesh.samples.regular_polygon(N,Name,Value,...)
```

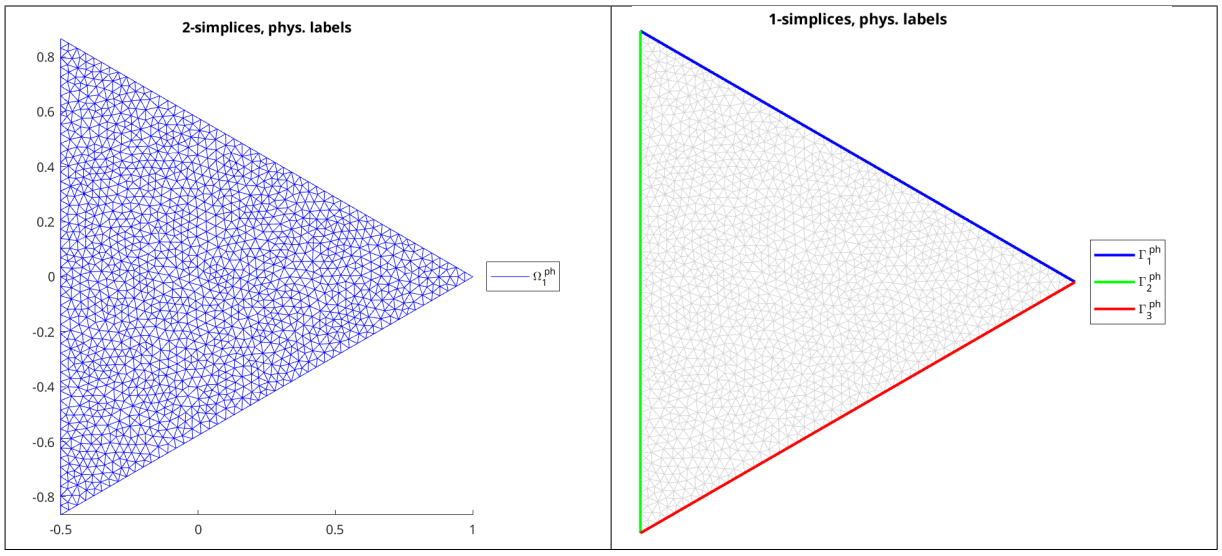
Description

fc_simesh.samples.regular_polygon(N) returns the regular polygon, centered in $[0,0]$ with $d=3$ points (so d faces) on the circle of radius $R=1$, as a siMesh object where N is a refinement parameter.

fc_simesh.samples.regular_polygon(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

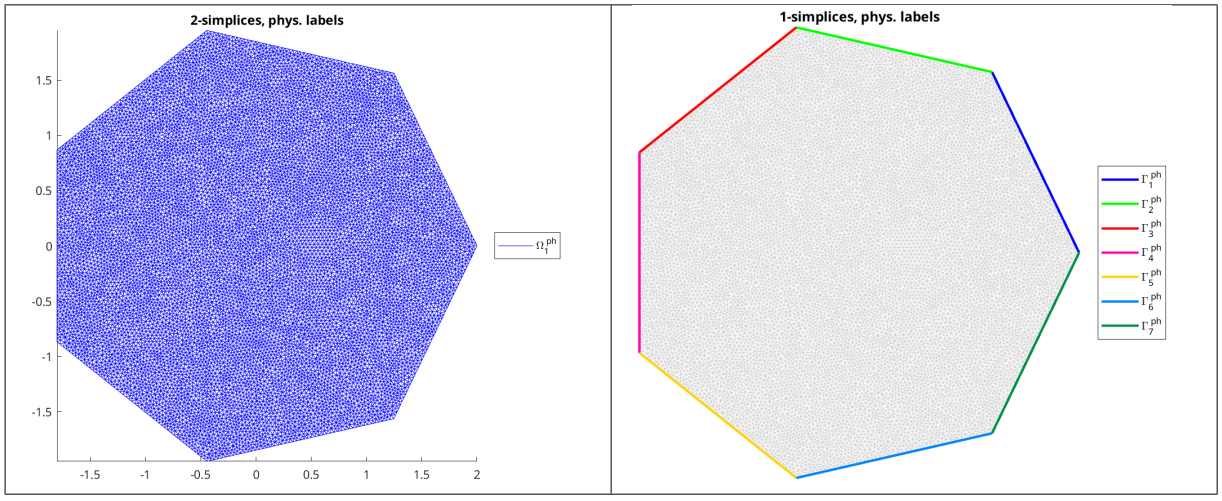
- 'R' : to specify the radius R (default 1),
- 'd' : to specify the number of points d (default 3),
- 'theta' : to specify the start point on the circle (default 0),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Figures 51 and 52.




```
Th=fc_simesh.samples.regular_polygon(50);
Th.plotmeshinfo()
```

Listing 51: siMesh object corresponding to the regular polygon with $d=3$ points and radius $R=1$.




```
Th=fc_simesh.samples.regular_polygon(50,'R',2,'d',7);
Th.plotmeshinfo()
```

Listing 52: siMesh object corresponding to the regular polygon with $d=7$ points and radius $R=2$.

4.1.7 fc_simesh.samples.disk5holes function

The **fc_simesh**.samples.disk5holes function returns an siMesh object corresponding to a disk with 5 holes obtain by using `gmsh` with `disk5holesOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.disk5holes(N)
Th=fc_simesh.samples.disk5holes(N,Name,Value, ...)
```

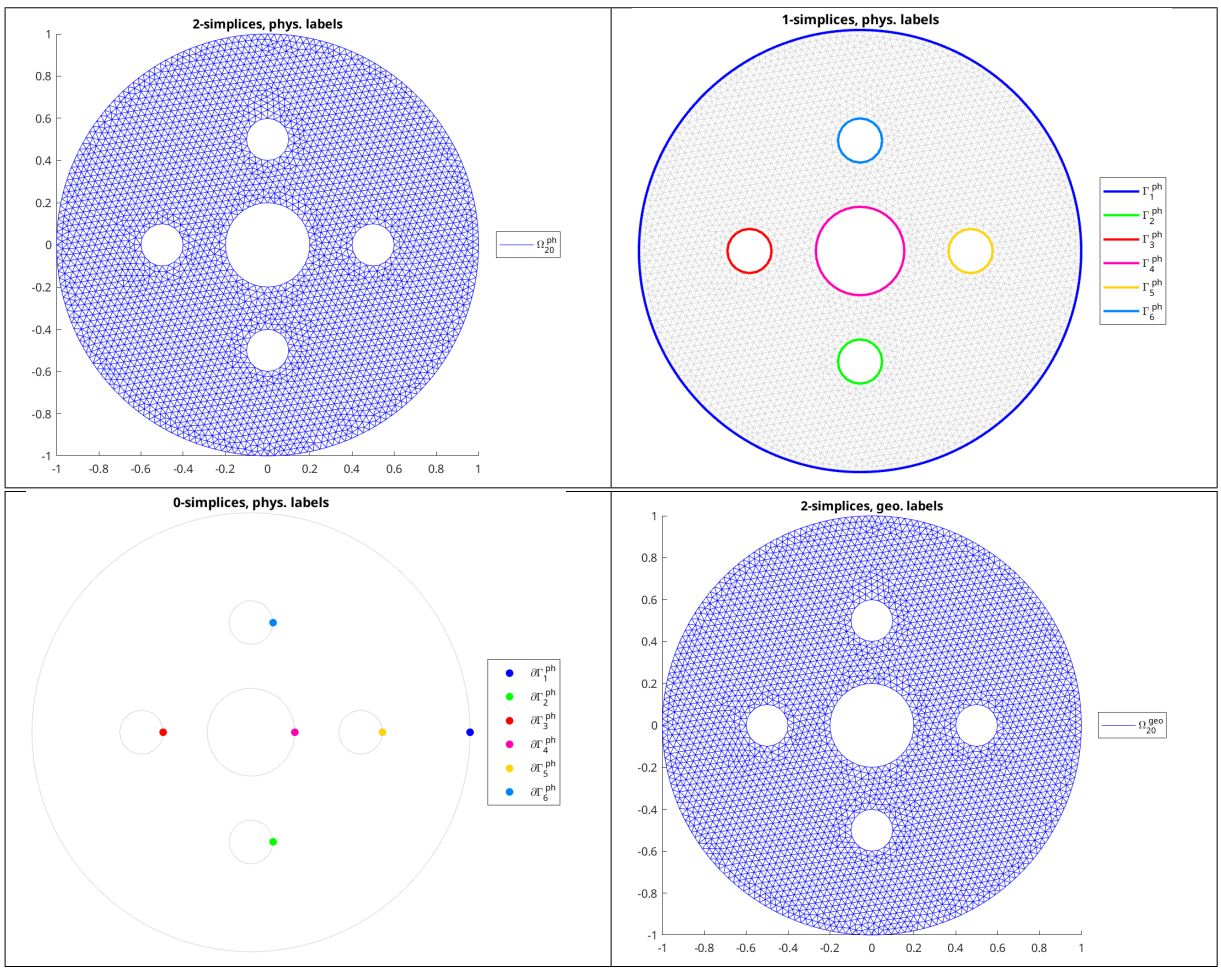
Description

fc_simesh.samples.disk5holes(N) returns the disk5holes, $\mathcal{C}([0,0],1)$, as a siMesh object where N is a refinement parameter.

fc_simesh.samples.disk5holes(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

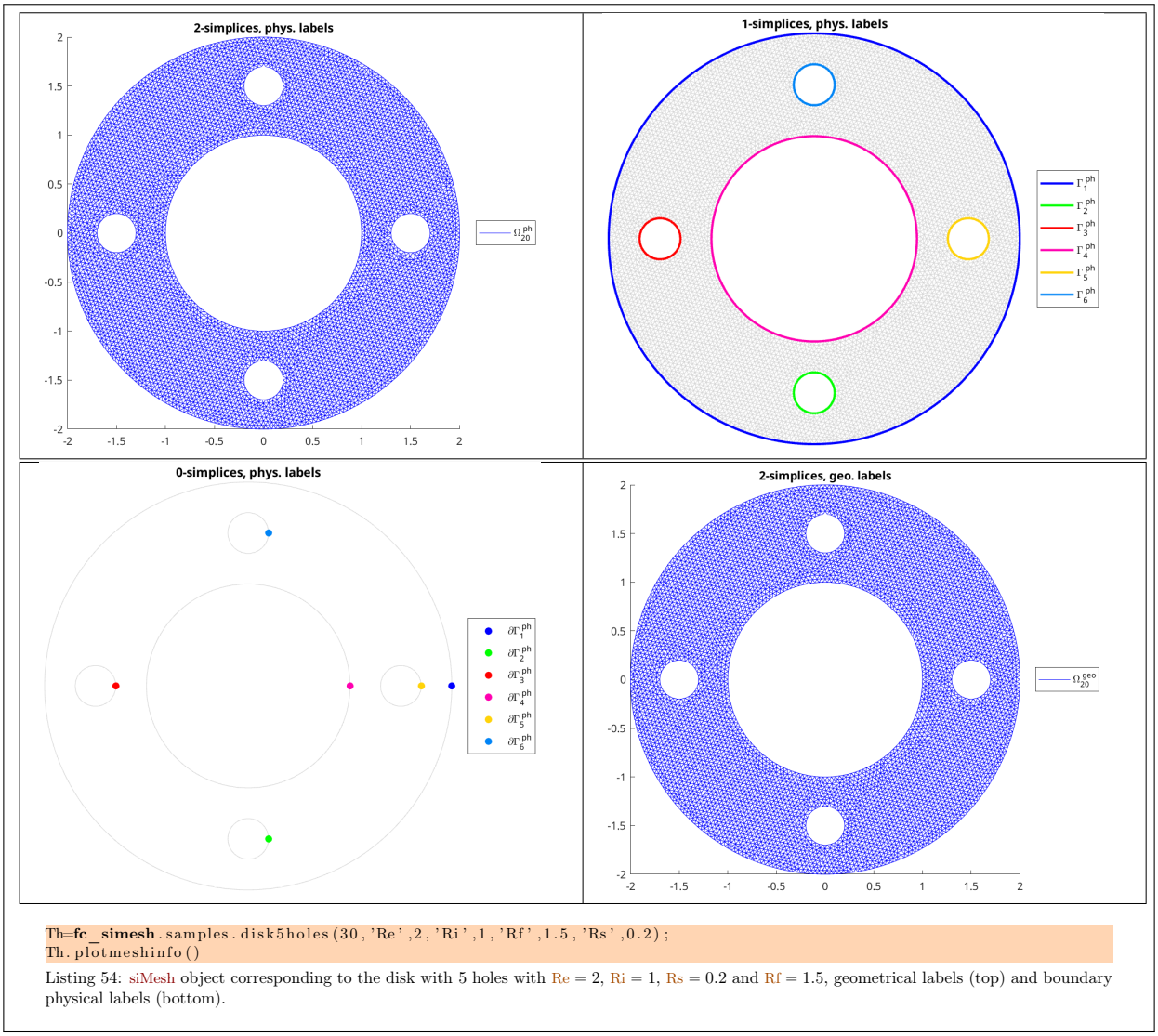
- 'Re' : radius of the centered exterior circle (default 1),
- 'Ri' : radius of the centered interior circle (default 0.2),
- 'Rs' : radius of the not centered small circles (default 0.1),
- 'Rf' : radius of the fictitious centered circle where center of small circles belong (default 0.5),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Listing 53.




```
Th=fc_simesh.samples.disk5holes(20);
Th.plotmeshinfo()
```

Listing 53: siMesh object corresponding to the disk with 5 holes with default parameters, geometrical labels (top) and boundary physical labels (bottom).




```
Th=fc_simesh.samples.disk5holes(30,'Re',2,'Ri',1,'Rf',1.5,'Rs',0.2);
Th.plotmeshinfo()
```

Listing 54: siMesh object corresponding to the disk with 5 holes with Re = 2, Ri = 1, Rs = 0.2 and Rf = 1.5, geometrical labels (top) and boundary physical labels (bottom).

4.1.8 fc_simesh.samples.red_cross function

The **fc_simesh**.samples.red_cross function returns an siMesh object corresponding the *red cross* symbol obtain by using `gmsh` with `red_cross0C.geo` file.

Syntaxe

```
Th=fc_simesh.samples.red_cross(N)
Th=fc_simesh.samples.red_cross(N,Name,Value,...)
```

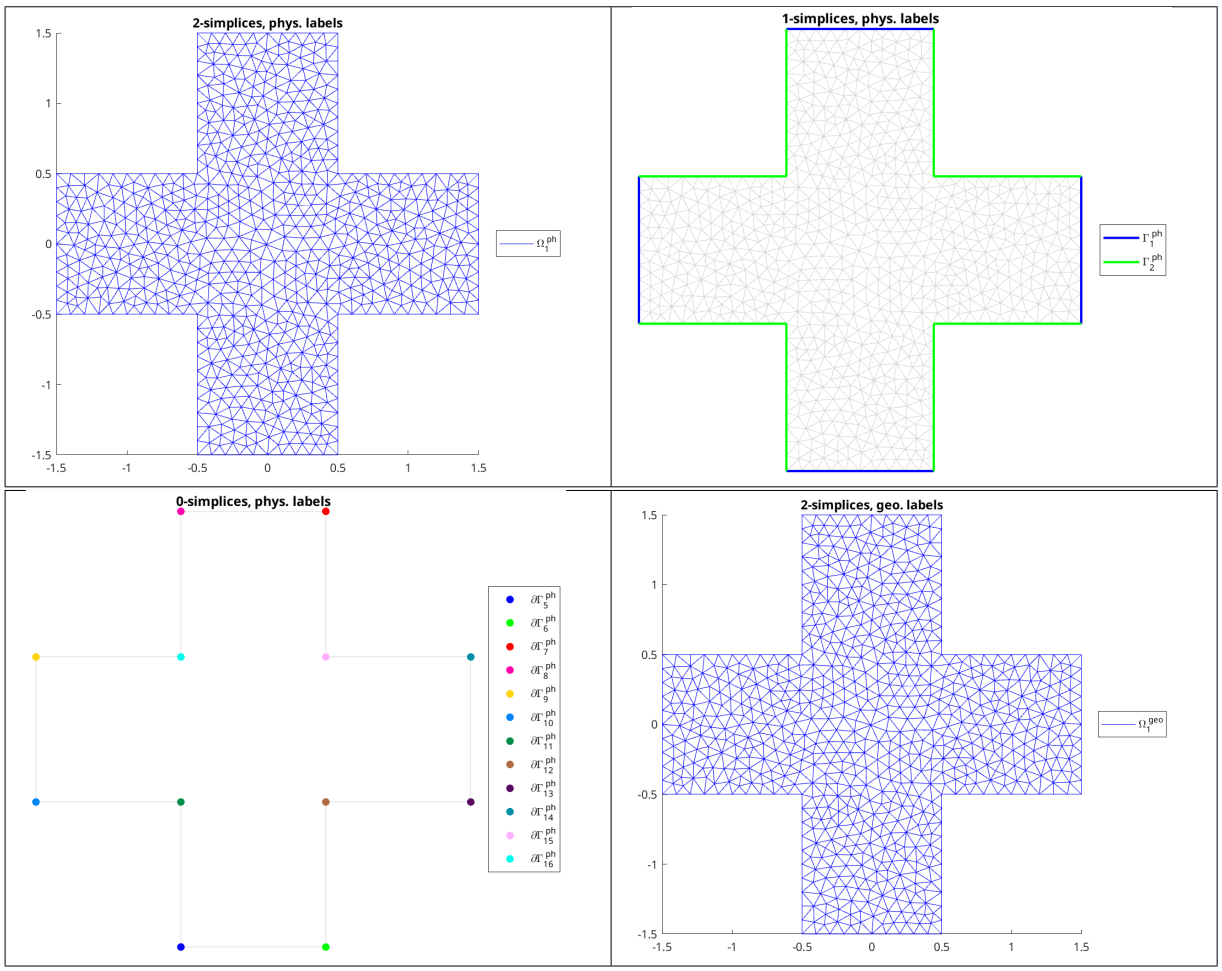
Description

fc_simesh.samples.red_cross(N) returns the red cross, centered in $[0,0]$ with global width $L = 3$ and arm width $h = 1$, as a siMesh object where N is a refinement parameter.

fc_simesh.samples.red_cross(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

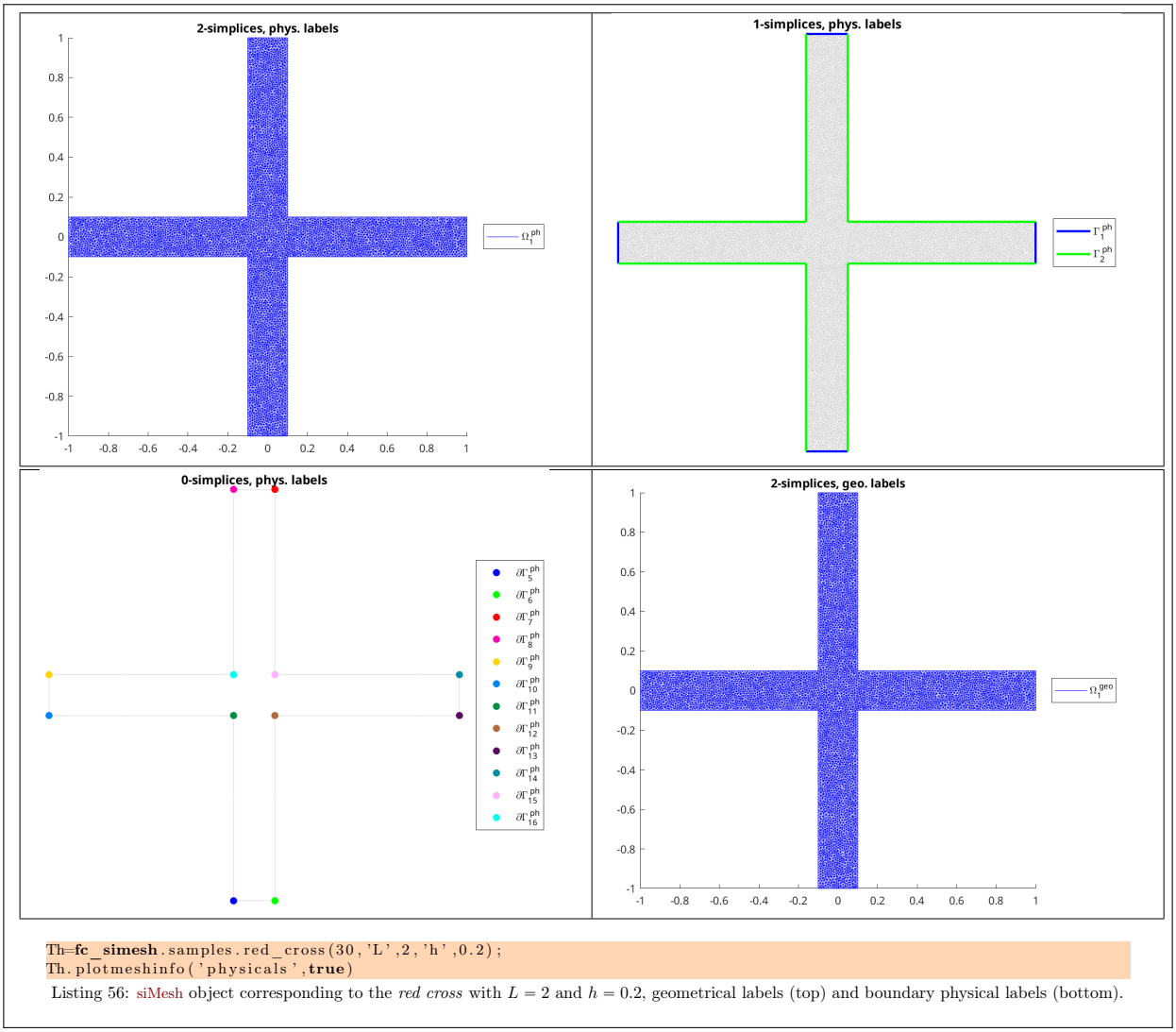
- 'L' : to specify the global width L (default 3),
- 'h' : to specify the arm width h (default 1),
- 'renumbering' : if **true**, physical tags are set to boundary elements (default **true**)
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Listings 55 and 56 for an example.




```
Th =fc_simesh.samples.red_cross(20);
Th.plotmeshinfo('physicals',true)
```

Listing 55: siMesh object corresponding to the *red cross* with default values ($L = 3$ and $h = 1$), geometrical labels (top) and boundary physical labels (bottom).




```
Th=fc_simesh.samples.red_cross(30,'L',2,'h',0.2);
Th.plotmeshinfo('physicals',true)
```

Listing 56: siMesh object corresponding to the *red cross* with $L = 2$ and $h = 0.2$, geometrical labels (top) and boundary physical labels (bottom).

4.1.9 fc_simesh.samples.model01 function

The **fc_simesh**.samples.model01 function returns an siMesh object obtain by using `gmsh` with `model01OC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.model01(N)
Th=fc_simesh.samples.model01(N,Name,Value,...)
```

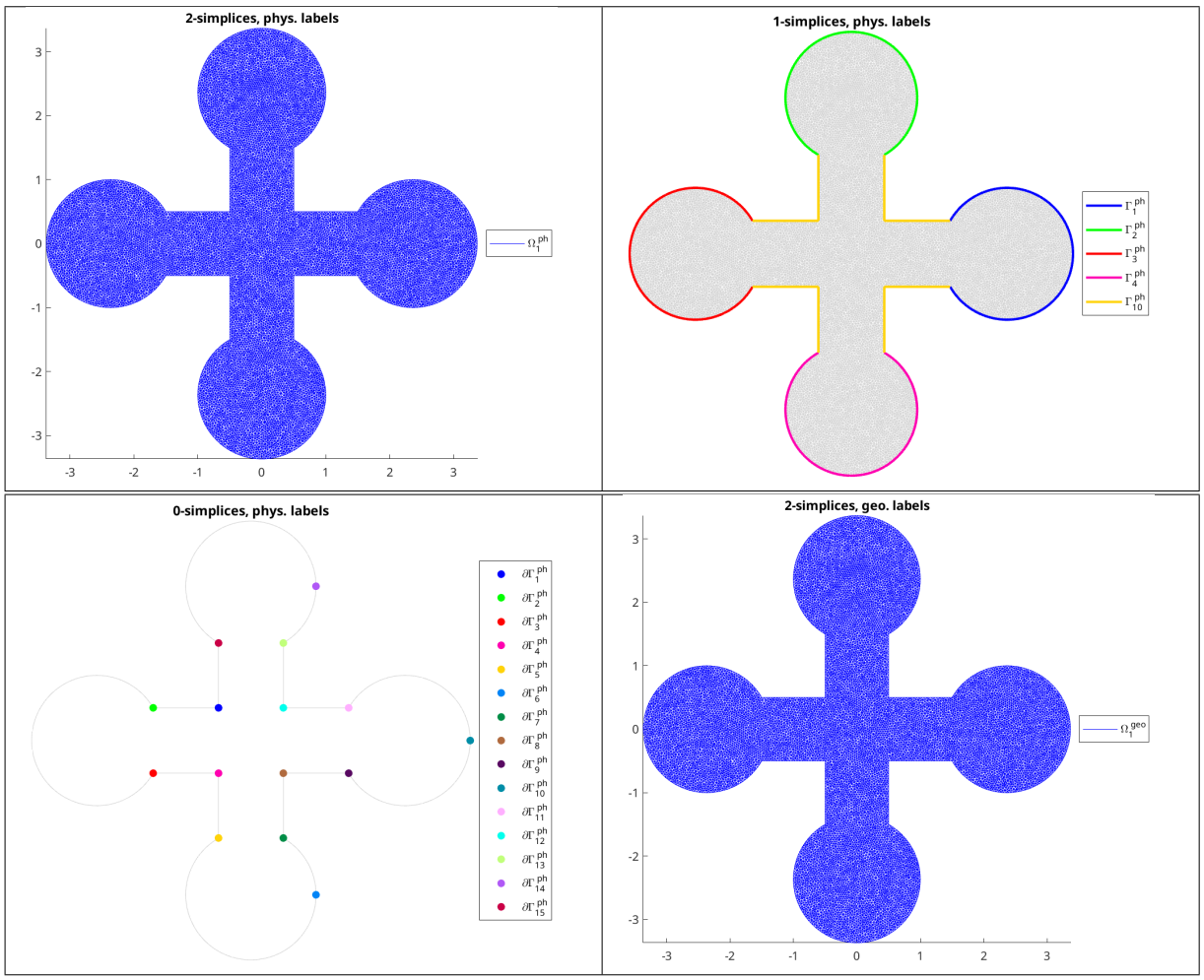
Description

fc_simesh.samples.model01(N) returns a siMesh object where N is a refinement parameter.

fc_simesh.samples.model01(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

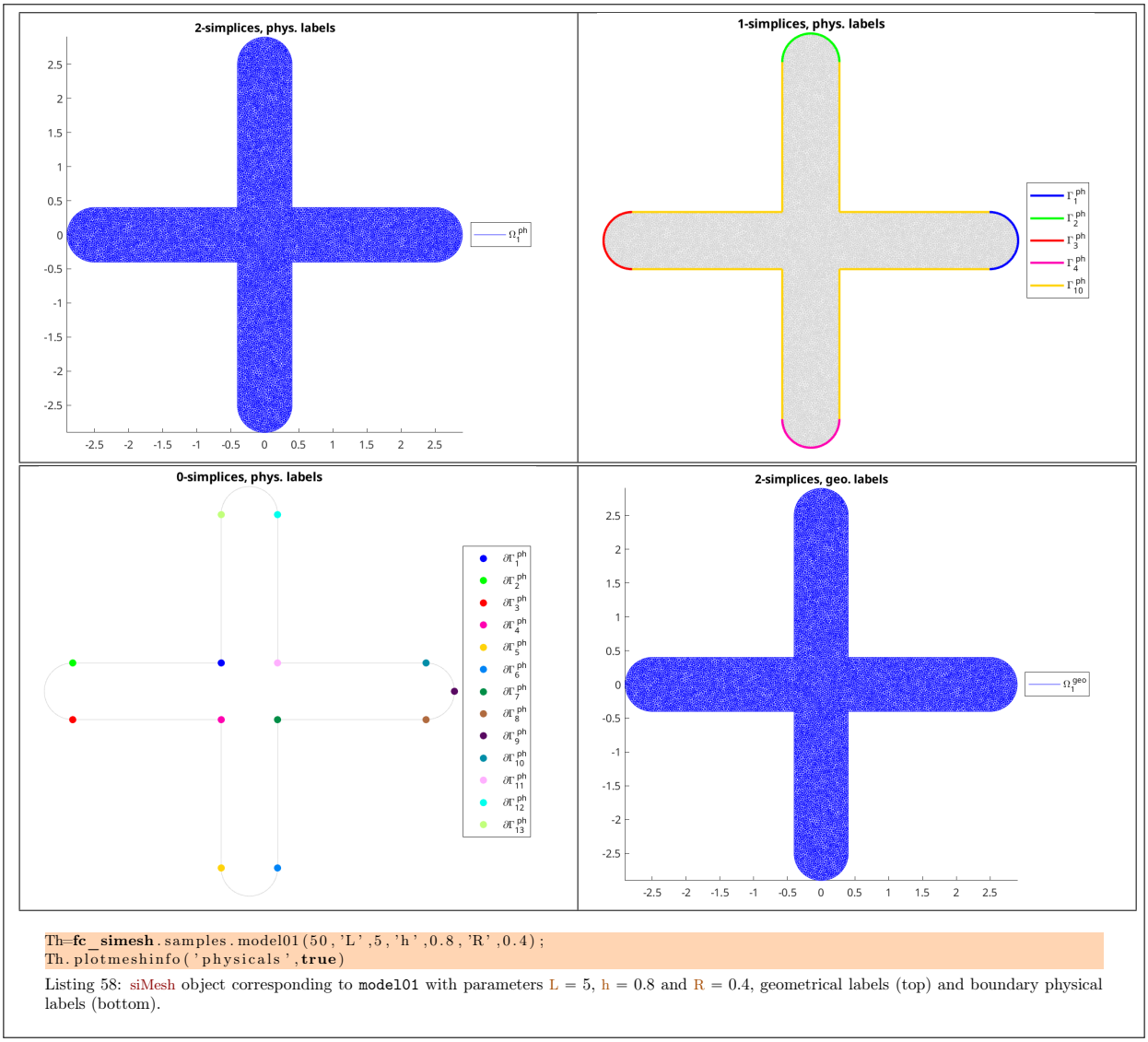
- 'L' : to specify the length L (default 3),
- 'h' : to specify the arm width h (default 1),
- 'R' : to specify the radius R (default 1),
- 'renumbering' : if **true**, physical tags are set to boundary elements (default **true**)
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Listings 57 and 58.




```
Th=fc_simesh.samples.model01(50);
Th.plotmeshinfo('physicals',true)
```

Listing 57: siMesh object corresponding to `model01` with default parameters, geometrical labels (top) and boundary physical labels (bottom).




```
Th=fc_simesh.samples.model01(50,'L',5,'h',0.8,'R',0.4);
Th.plotmeshinfo('physicals',true)
```

Listing 58: siMesh object corresponding to `model01` with parameters L = 5, h = 0.8 and R = 0.4, geometrical labels (top) and boundary physical labels (bottom).

4.1.10 fc_simesh.samples.model02 function

The **fc_simesh**.samples.model02 function returns an siMesh object obtain by using `gmsh` with `model02OC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.model02(N)
Th=fc_simesh.samples.model02(N,Name,Value,...)
```

Description

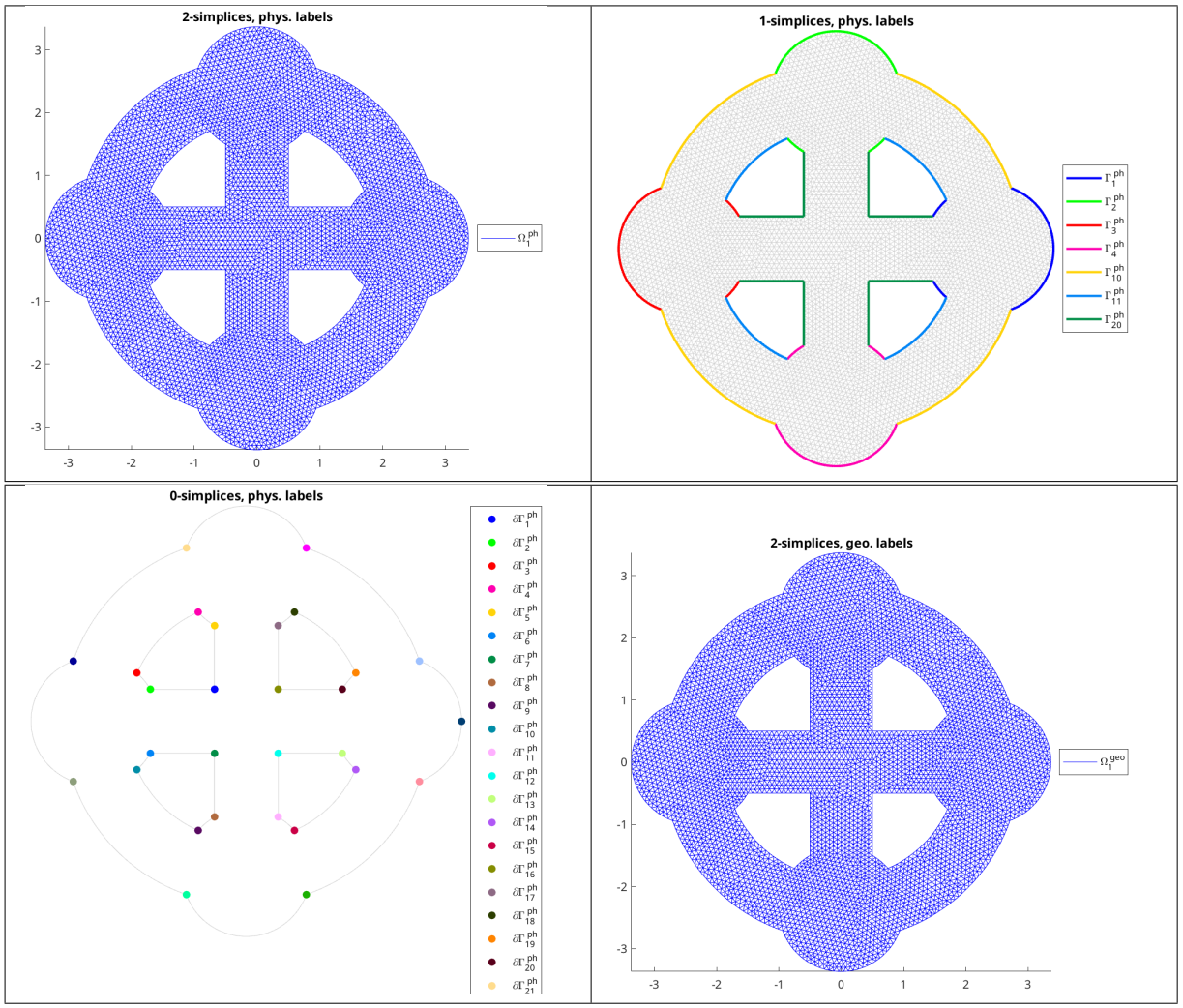
fc_simesh.samples.model02(N) returns a siMesh object where N is a refinement parameter.

fc_simesh.samples.model02(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

- 'L' : to specify the length L (default 3),
- 'h' : to specify the arm width h (default 1),
- 'R' : to specify the radius R (default 1),
- 'renumbering' : if **true**, physical tags are set to boundary elements (default **true**)
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

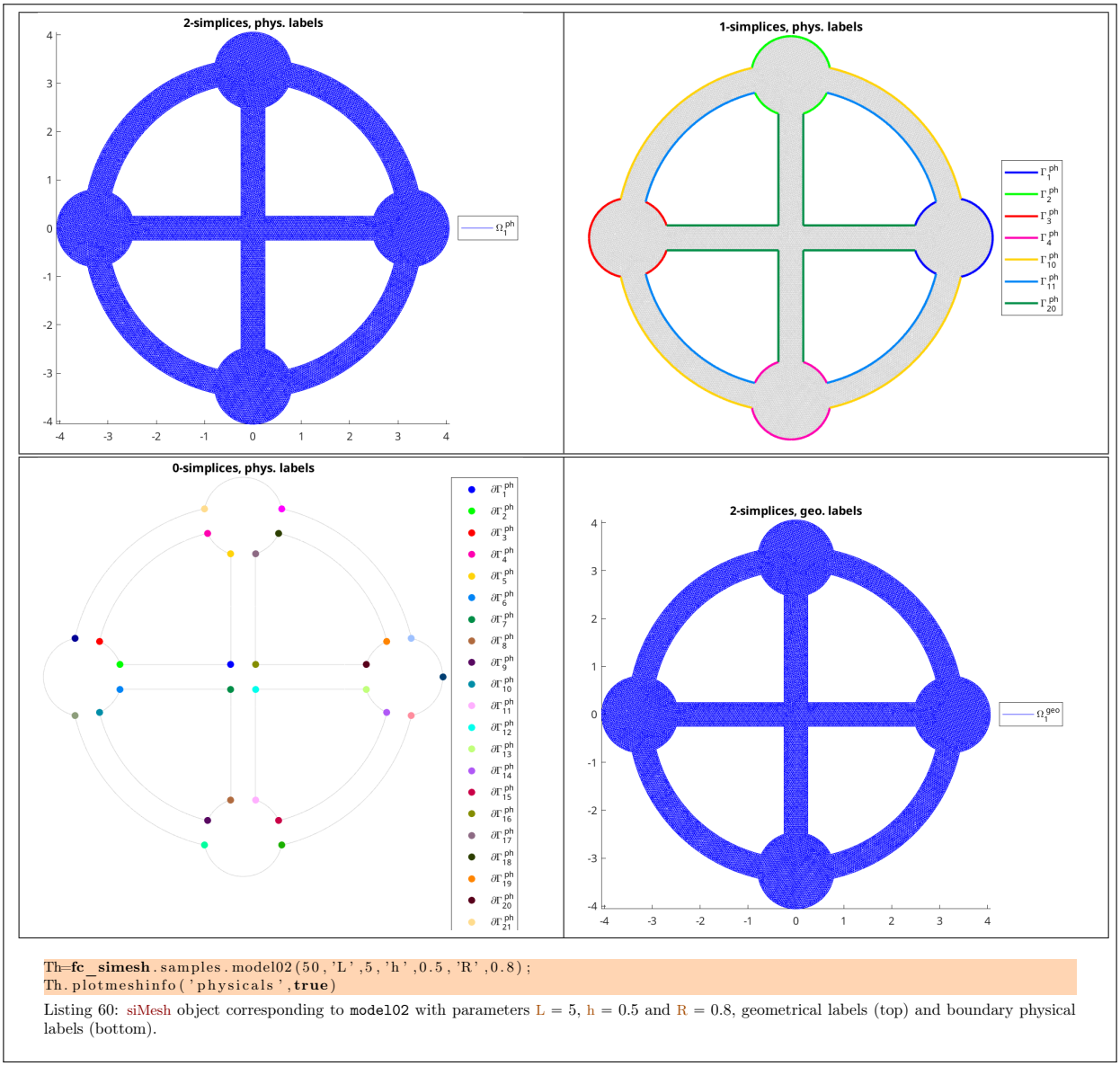
We must have $L/2 > R$ and $R > \sqrt{2}h/2$.

See Listings 59 and 60.




```
Th=fc_simesh.samples.model02(50);
Th.plotmeshinfo('physicals',true)
```

Listing 59: siMesh object corresponding to `model02` with default parameters, geometrical labels (top) and boundary physical labels (bottom).




```
Th=fc_simesh.samples.model02(50,'L',5,'h',0.5,'R',0.8);
Th.plotmeshinfo('physicals',true)
```

Listing 60: siMesh object corresponding to `model02` with parameters L = 5, h = 0.5 and R = 0.8, geometrical labels (top) and boundary physical labels (bottom).

4.1.11 fc_simesh.samples.model03 function

The **fc_simesh**.samples.model03 function returns an siMesh object obtain by using `gmsh` with `model03OC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.model03(N)
Th=fc_simesh.samples.model03(N,Name,Value,...)
```

Description

fc_simesh.samples.model03(N) returns a siMesh object where N is a refinement parameter.

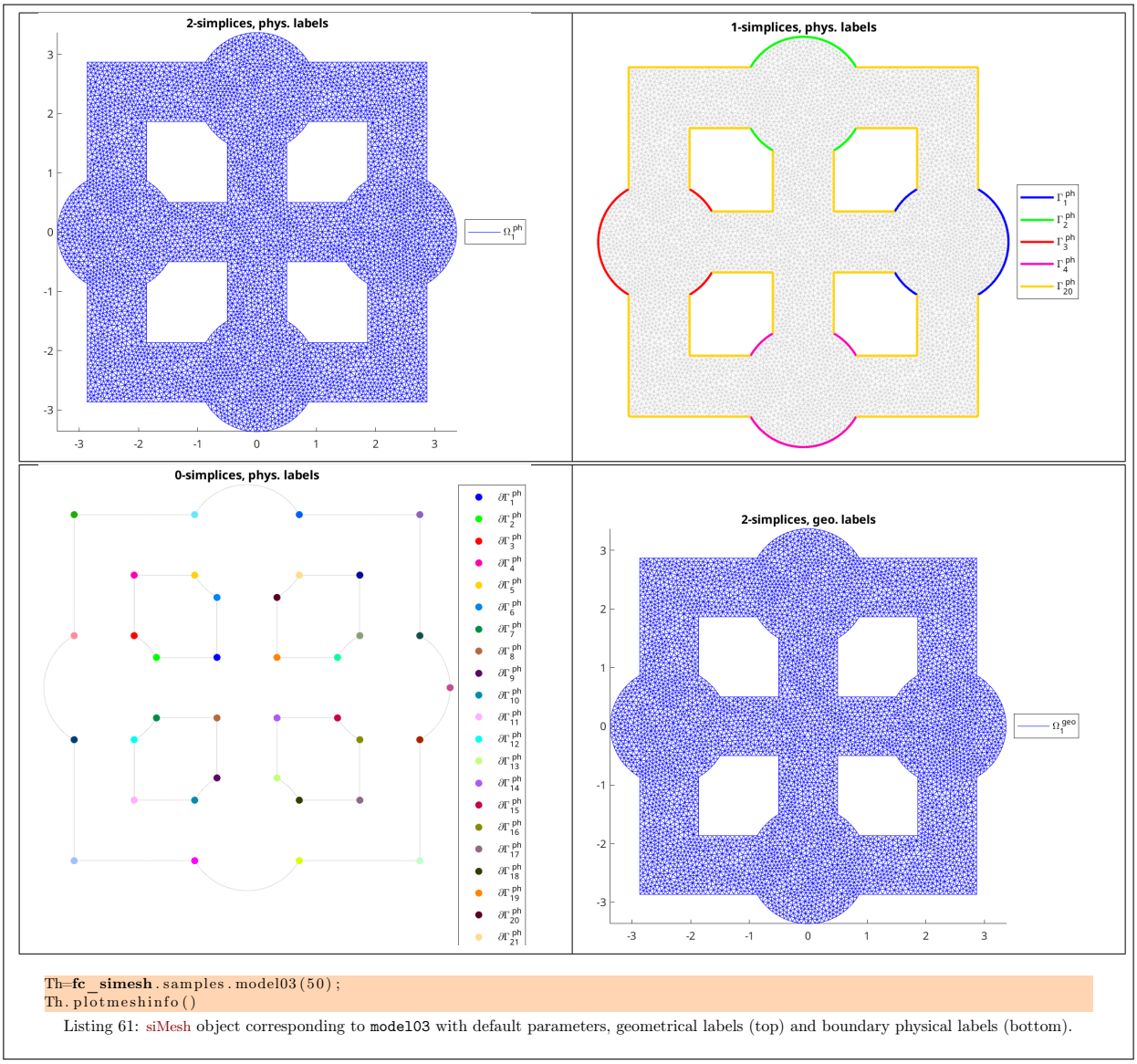
fc_simesh.samples.model03(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

- 'L' : to specify the length L (default 3),
- 'h' : to specify the arm width h (default 1),
- 'R' : to specify the radius R (default 1),
- 'renumbering' : if **true**, physical tags are set to boundary elements (default **true**)
- 'verbose' : to specify verbosity from 0 to 4...

- 'delete' : if true, deletes generated mesh file (default : **true**)

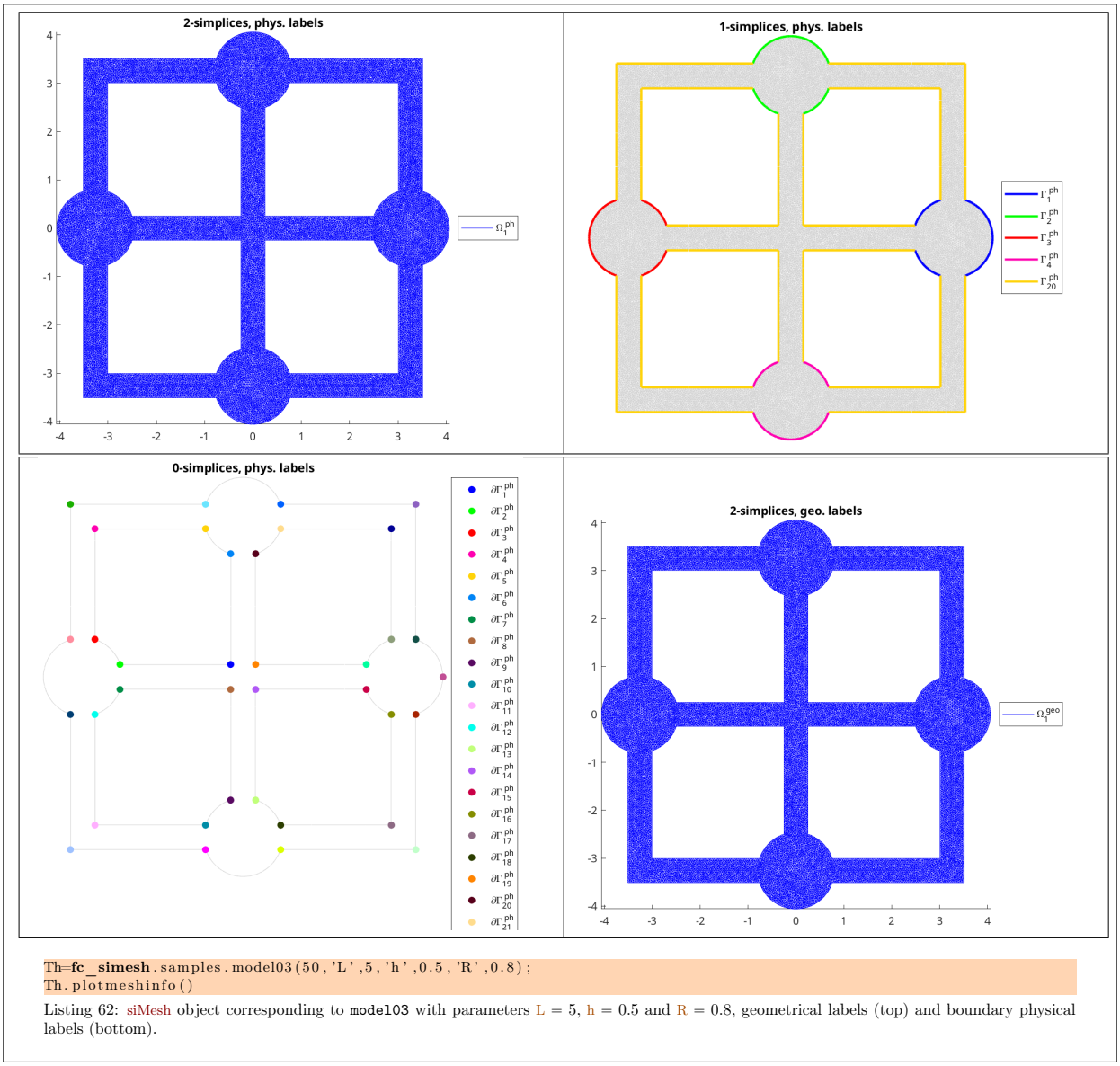
We must have $L/2 > R$ and $R > \sqrt{2}h/2$.

See Listings 61 and 62.




```
Th=fc_simesh.samples.model03(50);
Th.plotmeshinfo()
```

Listing 61: siMesh object corresponding to `model03` with default parameters, geometrical labels (top) and boundary physical labels (bottom).




```
Th=fc_simesh.samples.model03(50,'L',5,'h',0.5,'R',0.8);
Th.plotmeshinfo()
```

Listing 62: siMesh object corresponding to `model03` with parameters L = 5, h = 0.5 and R = 0.8, geometrical labels (top) and boundary physical labels (bottom).

4.1.12 fc_simesh.samples.model03v2 function

The **fc_simesh**.samples.model03v2 function returns an siMesh object obtain by using `gmsh` with `model03v20C.geo` file.

Syntaxe

```
Th=fc_simesh.samples.model03v2(N)
Th=fc_simesh.samples.model03v2(N,Name,Value,...)
```

Description

fc_simesh.samples.model03v2(N) returns a siMesh object where N is a refinement parameter.

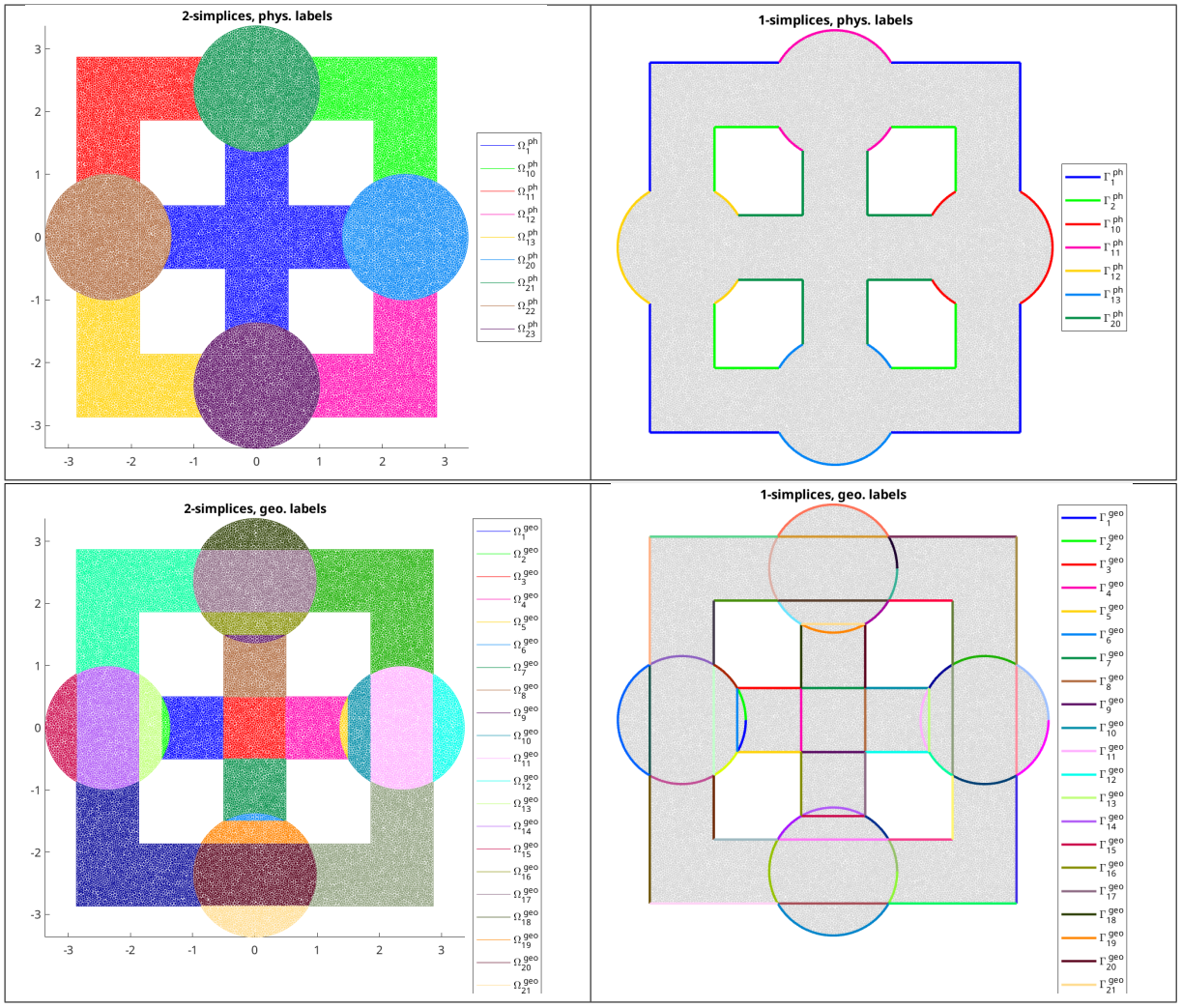
fc_simesh.samples.model03v2(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

- 'L' : to specify the length L (default 3),
- 'h' : to specify the arm width h (default 1),
- 'R' : to specify the radius R (default 1),
- 'renumbering' : if **true**, physical tags are set to boundary elements (default **true**)
- 'verbose' : to specify verbosity from 0 to 4...

- 'delete' : if true, deletes generated mesh file (default : **true**)

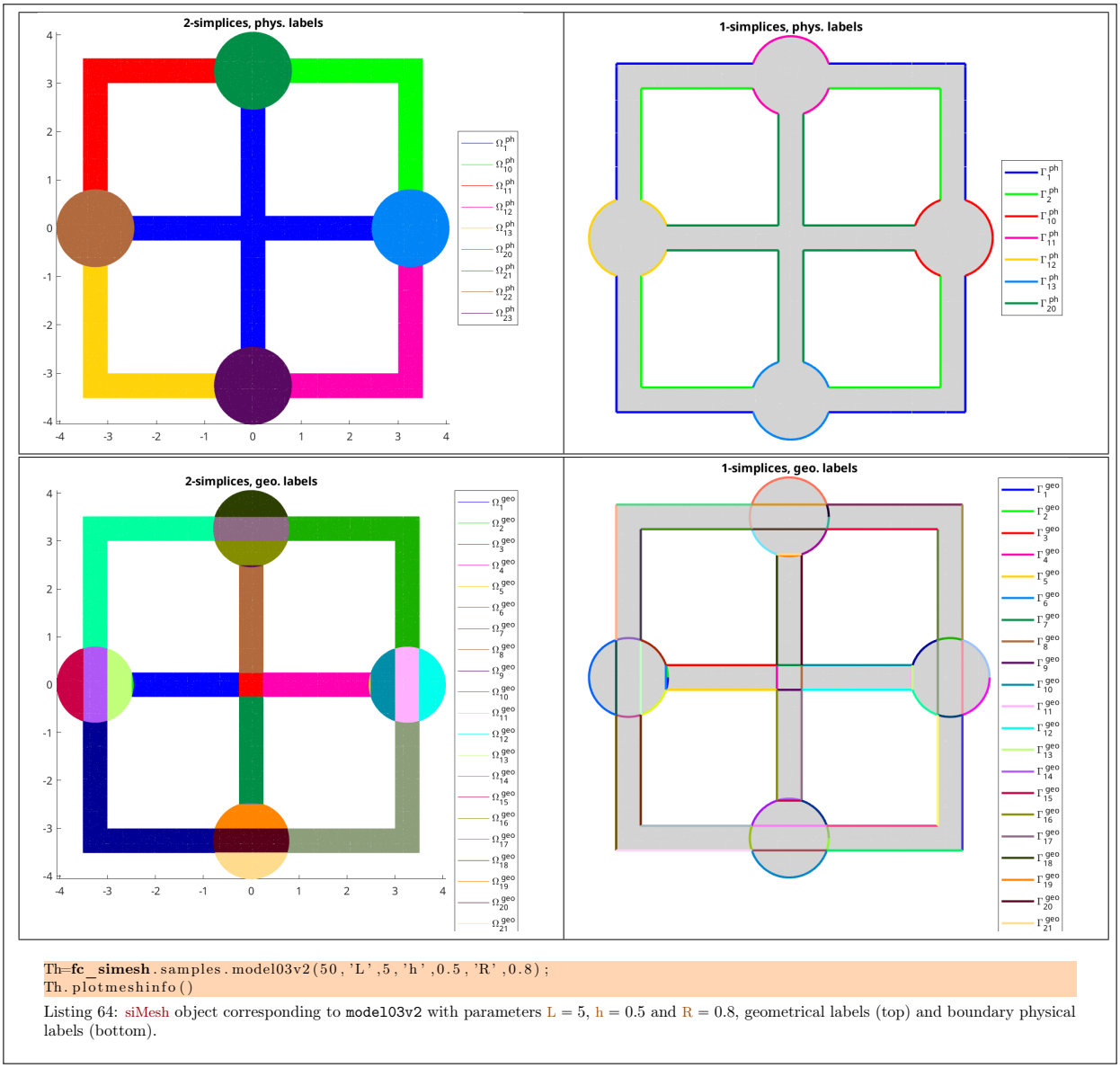
We must have $L/2 > R$ and $R > \sqrt{2}h/2$.

See Listings 63 and 64.




```
Th=fc_simesh.samples.model03v2(50);
Th.plotmeshinfo()
```

Listing 63: siMesh object corresponding to `model03v2` with default parameters, geometrical labels (top) and boundary physical labels (bottom).




```
Th=fc_simesh.samples.model03v2(50,'L',5,'h',0.5,'R',0.8);
Th.plotmeshinfo()
```

Listing 64: siMesh object corresponding to `model03v2` with parameters L = 5, h = 0.5 and R = 0.8, geometrical labels (top) and boundary physical labels (bottom).

4.1.13 fc_simesh.samples.olympic_rings function

The **fc_simesh**.samples.olympic_rings function returns an siMesh object obtain by using `gmsh` with `olympic_ringsOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.olympic_rings(N)
Th=fc_simesh.samples.olympic_rings(N,Name,Value,...)
```

Description

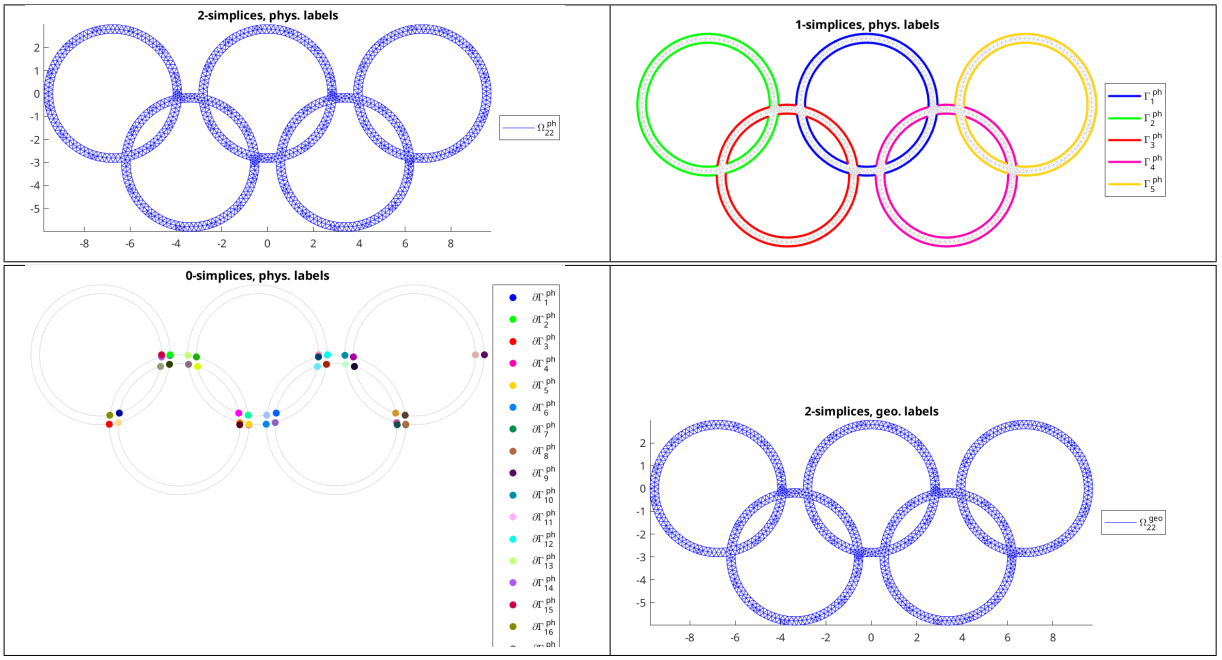
fc_simesh.samples.olympic_rings(N) returns a siMesh object where N is a refinement parameter.

fc_simesh.samples.olympic_rings(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

- 'h' : to specify the thickness h of the rings (default 0.5),
- 'R' : to specify the radius R of the rings (default 3),
- 'renumbering' : if **true**, physical tags are set to domain elements and boundary elements (default **true**)
- 'verbose' : to specify verbosity from 0 to 4...

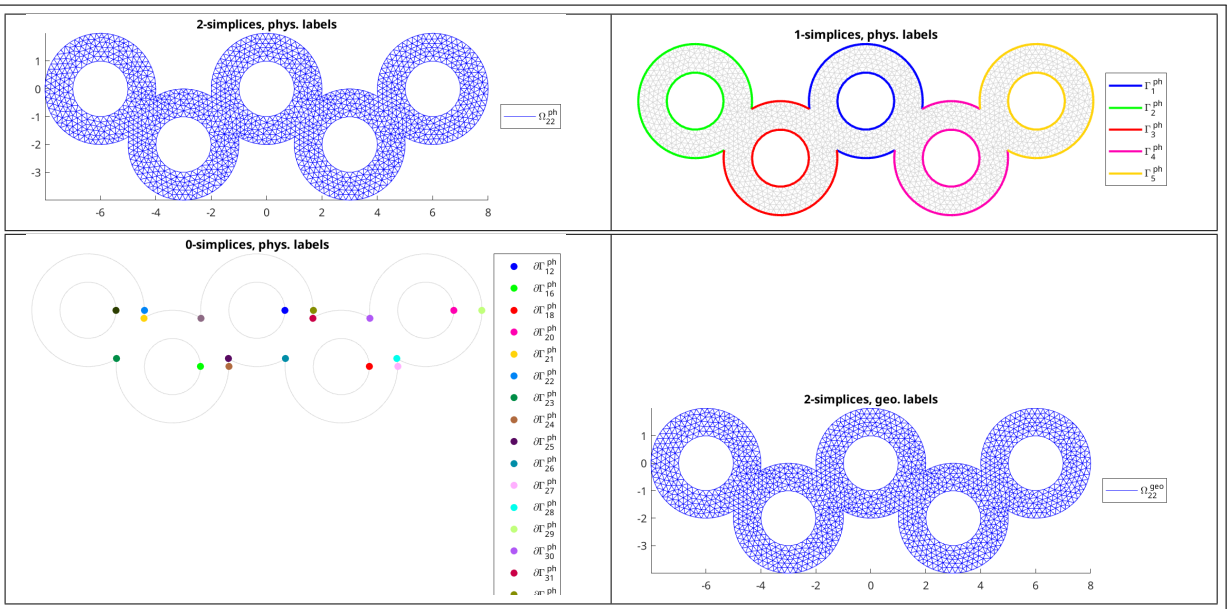
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Listings 65 and 66.




```
Th=fc_simesh.samples.olympic_rings(50);
Th.plotmeshinfo('physicals',true)
```

Listing 65: siMesh object corresponding to `olympic_rings` with default parameters, geometrical labels (top) and physical labels (bottom).




```
Th=fc_simesh.samples.olympic_rings(30,'R',2,'h',2);
Th.plotmeshinfo('physicals',true)
```

Listing 66: siMesh object corresponding to `olympic_rings` with parameters R = 2 and h = 2, geometrical labels (top) and physical labels (bottom).

4.1.14 fc_simesh.samples.olympic_ringsv2 function

The **fc_simesh**.samples.olympic_ringsv2 function returns an siMesh object obtain by using `gmsh` with `olympic_ringsv20C.geo` file.

Syntaxe

```
Th=fc_simesh.samples.olympic_ringsv2(N)
Th=fc_simesh.samples.olympic_ringsv2(N,Name,Value,...)
```

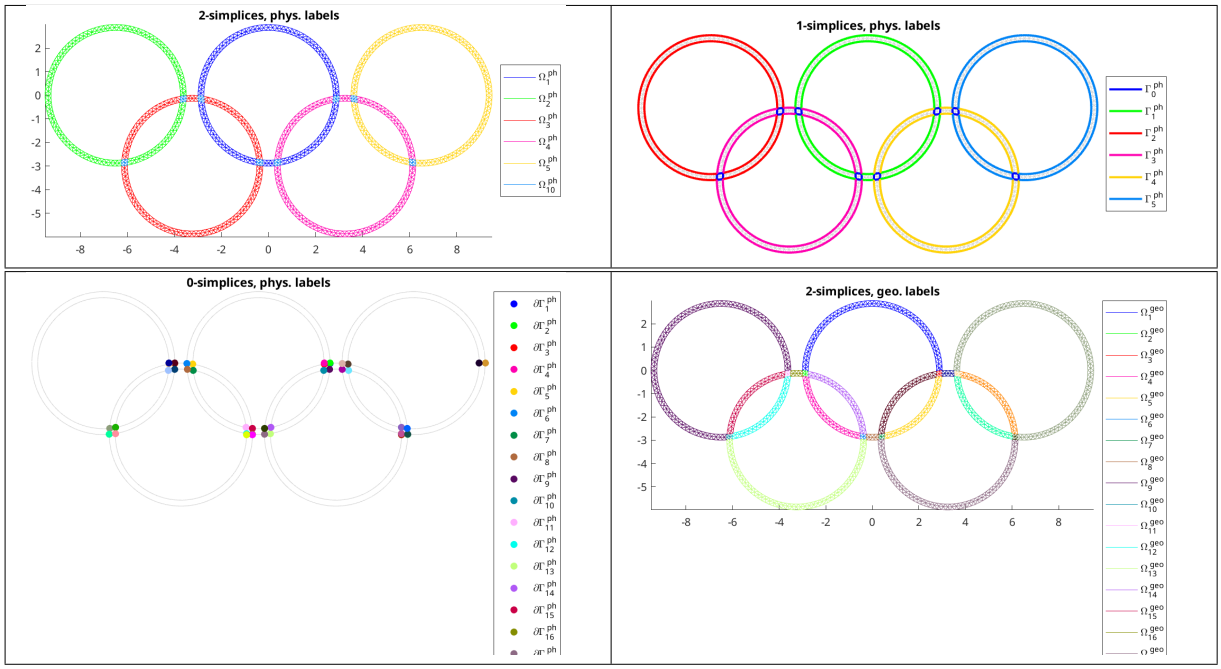
Description

fc_simesh.samples.olympic_ringsv2(N) returns a siMesh object where N is a refinement parameter.

fc_simesh.samples.olympic_ringsv2(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

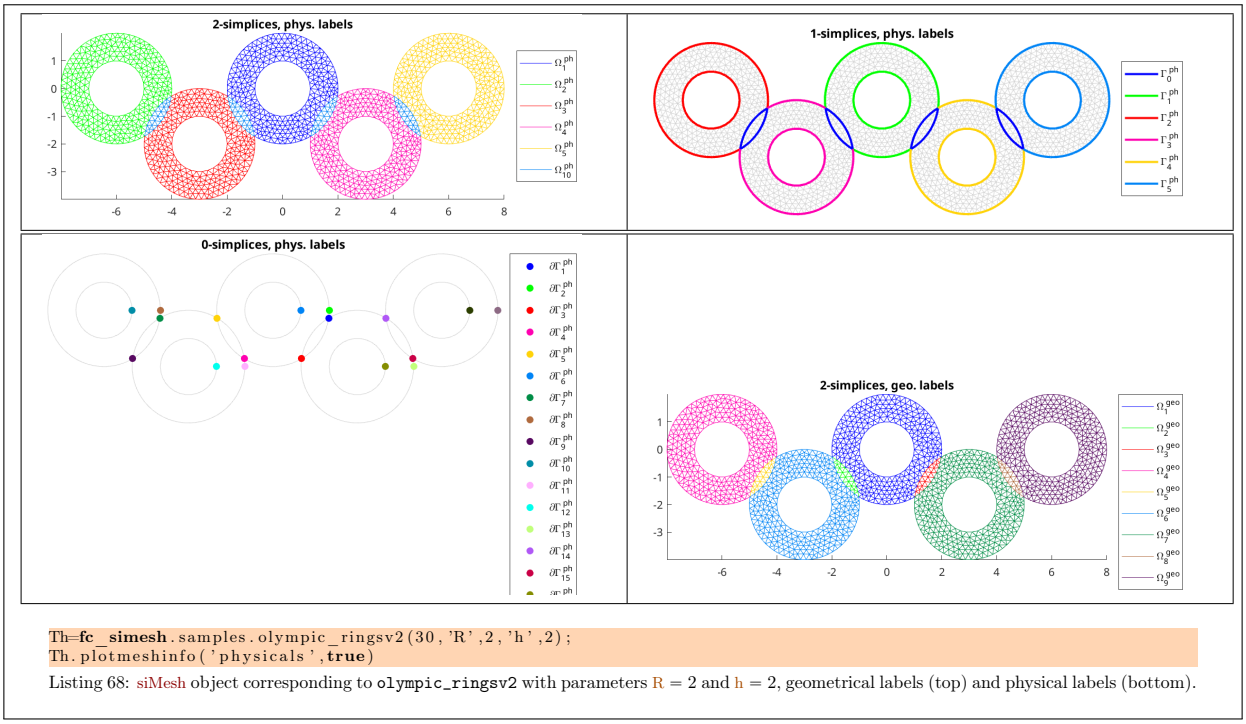
- 'h' : to specify the thickness h of the ringsv2 (default 0.5),
- 'R' : to specify the radius R of the ringsv2 (default 3),
- 'renumbering' : if **true**, physical tags are set to domain elements and boundary elements (default **true**)
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Listings 67 and 68.




```
Th=fc_simesh.samples.olympic_ringsv2(50);
Th.plotmeshinfo('physicals',true)
```

Listing 67: siMesh object corresponding to `olympic_ringsv2` with default parameters, geometrical labels (top) and physical labels (bottom).




```
Th=fc_simesh.samples.olympic_ringsv2(30,'R',2,'h',2);
Th.plotmeshinfo('physicals',true)
```

Listing 68: siMesh object corresponding to `olympic_ringsv2` with parameters R = 2 and h = 2, geometrical labels (top) and physical labels (bottom).

4.1.15 fc_simesh.samples.model10 function

The **fc_simesh**.samples.model10 function returns an siMesh object obtain by using `gmsh` with `model10OC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.model10(N)
Th=fc_simesh.samples.model10(N,Name,Value,...)
```

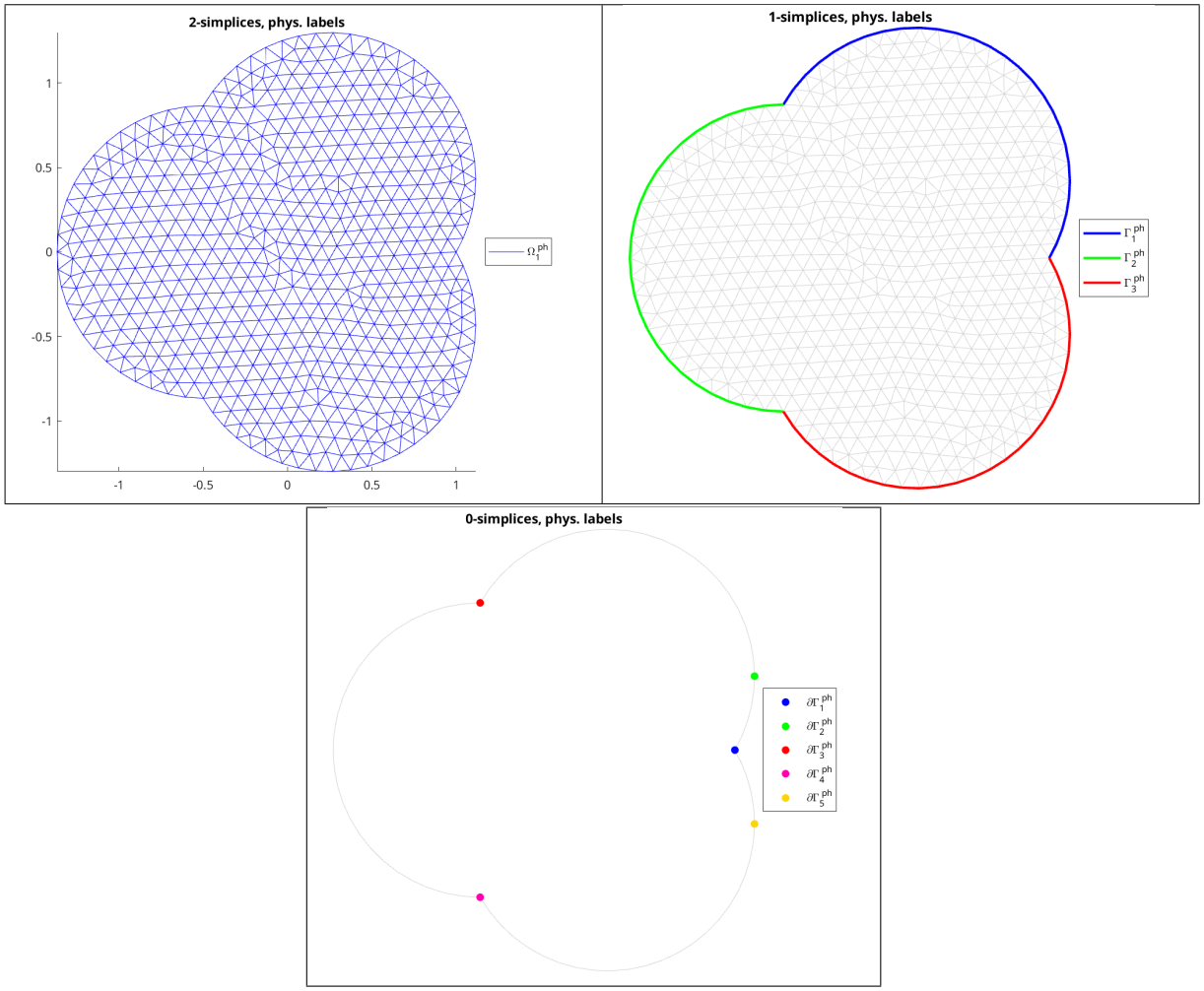
Description

fc_simesh.samples.model10(N) returns a siMesh object where N is a refinement parameter.

fc_simesh.samples.model10(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

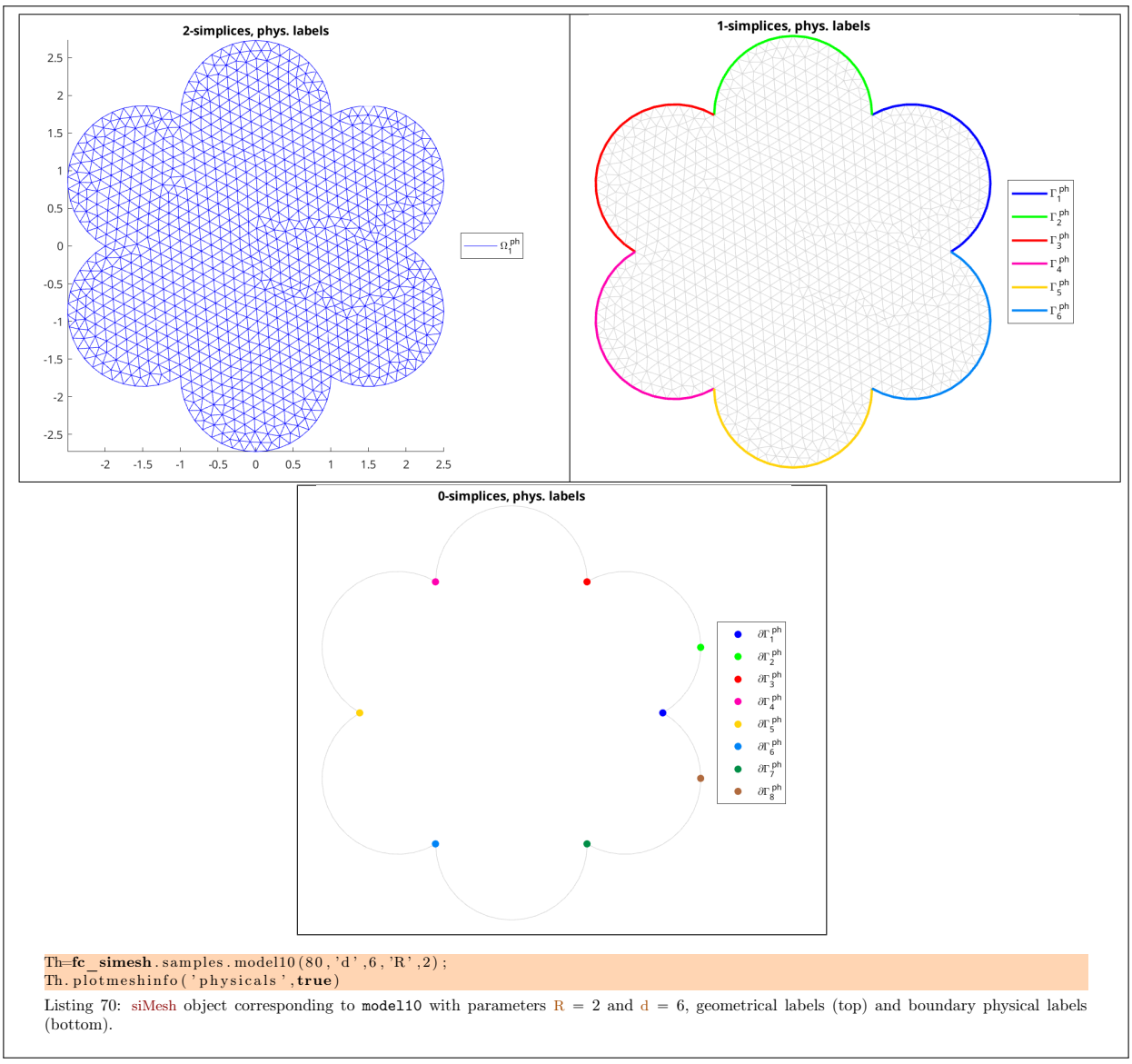
- 'R' : to specify the radius R (default 1),
- 'd' : to specify the number of points d (default 3),
- 'theta' : to specify the start point on the circle (default 0),
- 'renumbering' : if **true**, physical tags are set to boundary elements (default **true**)
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Listings 69 and 70.




```
Th=fc_simesh.samples.model10(60);
Th.plotmeshinfo('physicals',true)
```

Listing 69: siMesh object corresponding to `model10` with default parameters, geometrical labels (top) and boundary physical labels (bottom).




```
Th=fc_simesh.samples.model10(80,'d',6,'R',2);
Th.plotmeshinfo('physicals',true)
```

Listing 70: siMesh object corresponding to `model10` with parameters R = 2 and d = 6, geometrical labels (top) and boundary physical labels (bottom).

4.1.16 fc_simesh.samples.model11 function

The **fc_simesh**.samples.model11 function returns an siMesh object obtain by using `gmsh` with `model11OC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.model11(N)
Th=fc_simesh.samples.model11(N,Name,Value,...)
```

Description

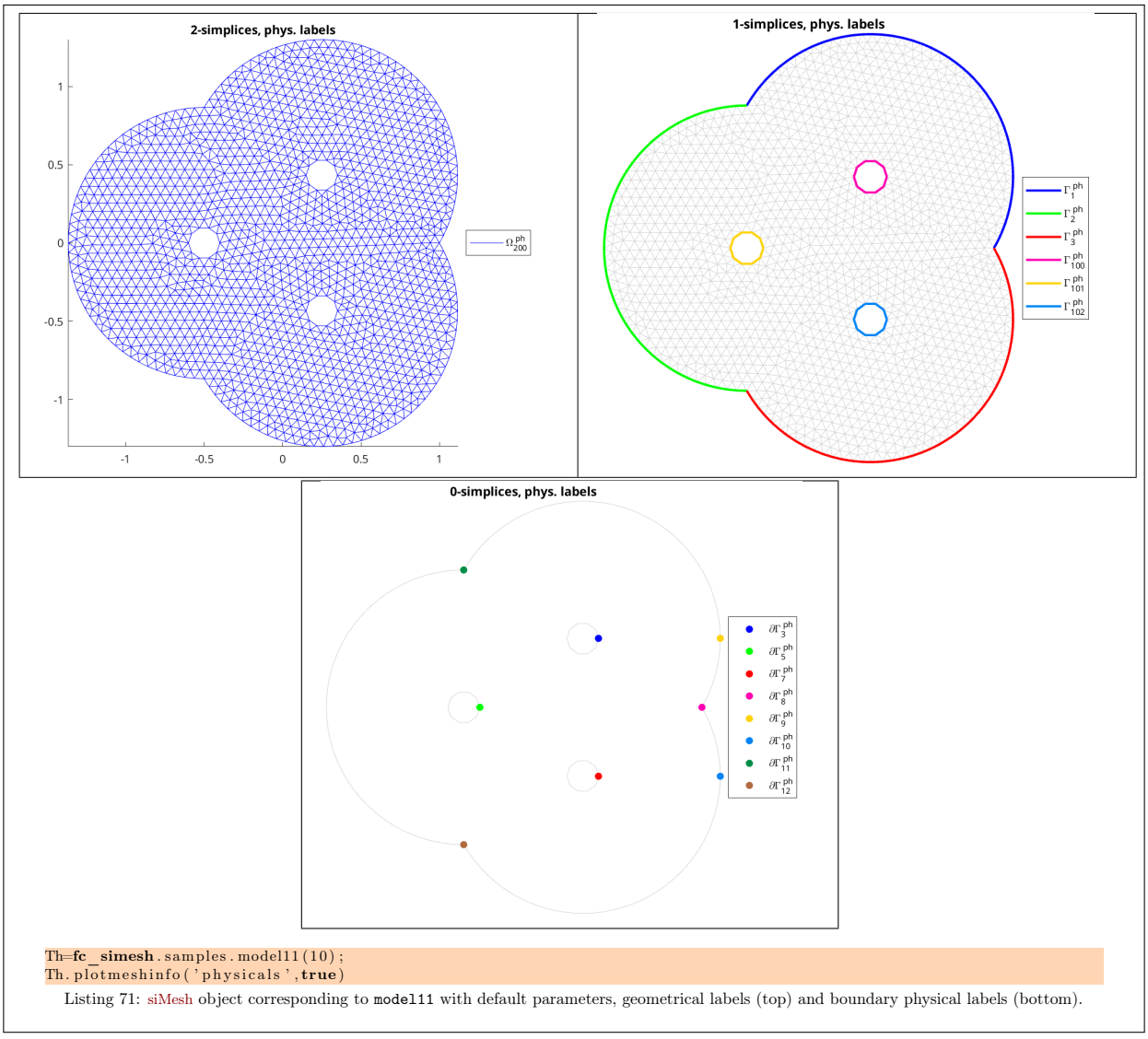
fc_simesh.samples.model11(N) returns a siMesh object where N is a refinement parameter.

fc_simesh.samples.model11(N,Name,Value,...) specifies function options using one or more Name,Value pair arguments. Options are

- 'R' : to specify the radius R (default 1),
- 'ri' : to specify the radius r_i of d small disk holes (default 0.1),
- 'd' : to specify the number of points d (default 3),
- 'theta' : to specify the start point on the circle (default 0),
- 'renumbering' : if **true**, physical tags are set to boundary elements (default **true**)

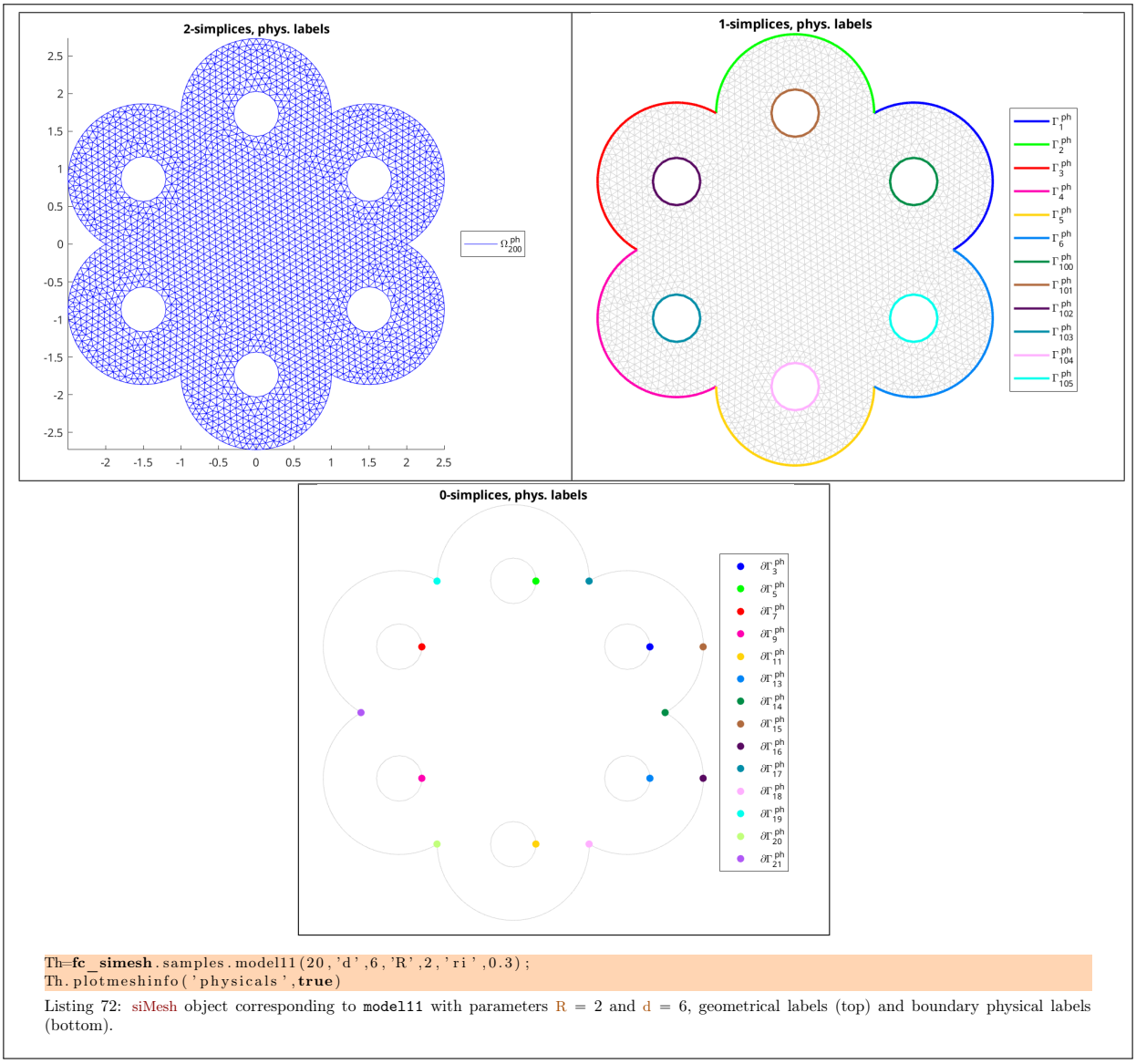
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

See Listings 71 and 72.




```
Th=fc_simesh.samples.model11(10);
Th.plotmeshinfo('physicals',true)
```

Listing 71: siMesh object corresponding to `model11` with default parameters, geometrical labels (top) and boundary physical labels (bottom).




```
Th=fc_simesh.samples.model11(20,'d',6,'R',2,'ri',0.3);
Th.plotmeshinfo('physicals',true)
```

Listing 72: siMesh object corresponding to `model11` with parameters R = 2 and d = 6, geometrical labels (top) and boundary physical labels (bottom).

4.2 3-simplicial and 2-simplicial meshes in $\mathbb{R}^3$

4.2.1 fc_simesh.samples.box function

The **fc_simesh**.samples.box function returns an siMesh object corresponding to a box obtain by using `gmsh` with `boxOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.box(N)
Th=fc_simesh.samples.box(N,Name,Value, ...)
```

Description

fc_simesh.samples.box(N) returns a box as a siMesh object where N is a refinement parameter. By default, the box is

$$[X, X + DX] \times [Y, Y + DY] \times [Z, Z + DZ]$$

where X = 0, Y = 0, Z = 0, DX = 1,DY = 1 and DZ = 1.

fc_simesh.samples.box(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, d ∈ {2, 3}, volume mesh (3 default) or surface mesh (2),

- 'X' : to specify X value (default 0),
- 'Y' : to specify Y value (default 0),
- 'Z' : to specify Z value (default 0),
- 'DX' : to specify DX value (default 1),
- 'DY' : to specify DY value (default 1),
- 'DZ' : to specify DZ value (default 1),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.box(10);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 5 and in Figure 6.

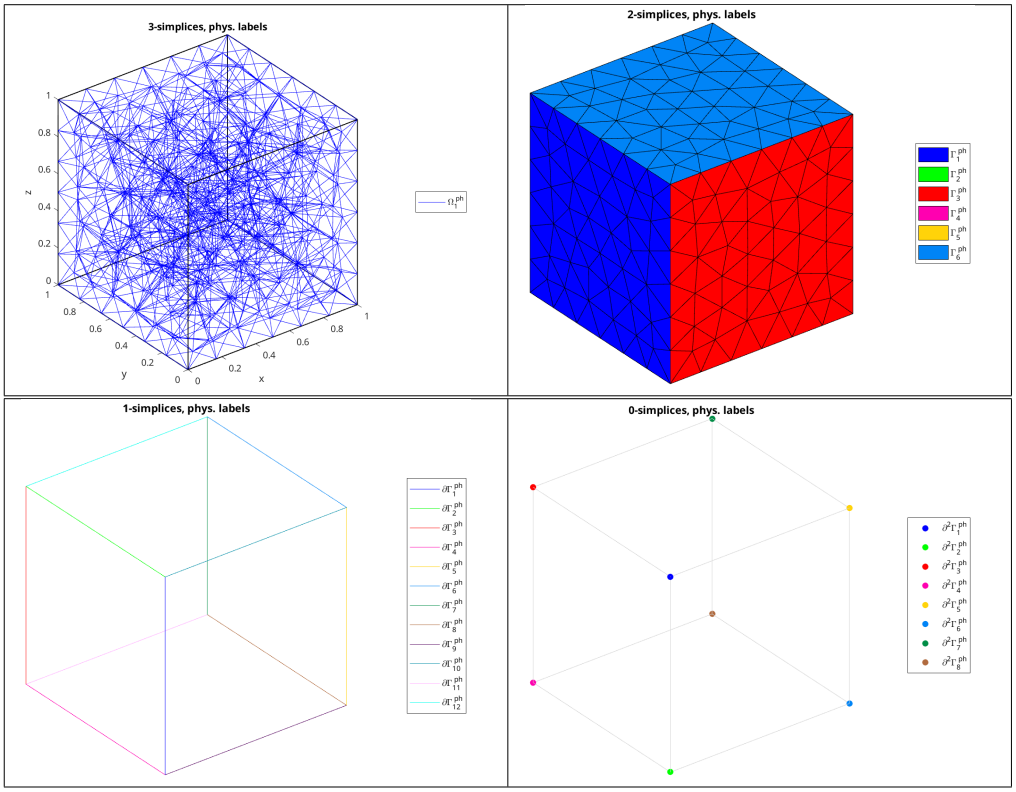


Figure 5: siMesh object corresponding to the default box (i.e. the unit cube). Representation of the physical labels.

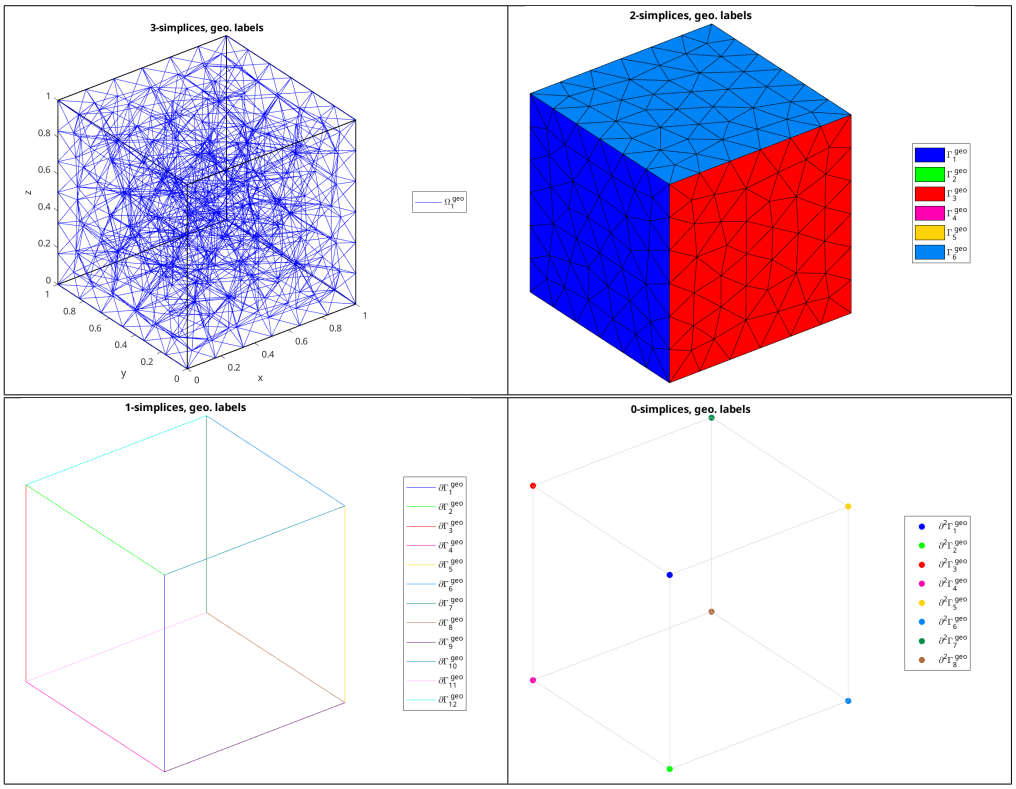


Figure 6: siMesh object corresponding to the default box (i.e. the unit cube). Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.box(10,'X',-1,'Y',-1,'Z',0,'DX',2,'DY',2,'DZ',3);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 7 and in Figure 8.

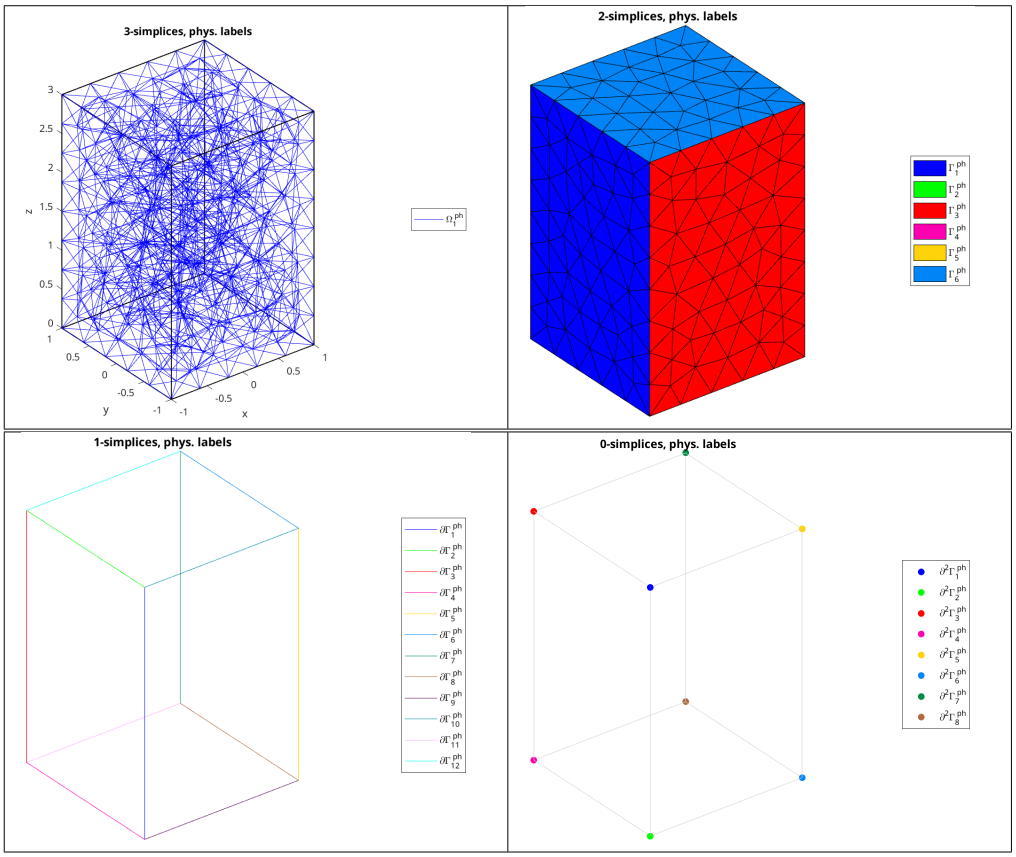


Figure 7: siMesh object corresponding to the box with parameters X = −1, Y = −1, Z = 0, DX = 2, DY = 2, DZ = 3. Representation of the physical labels.

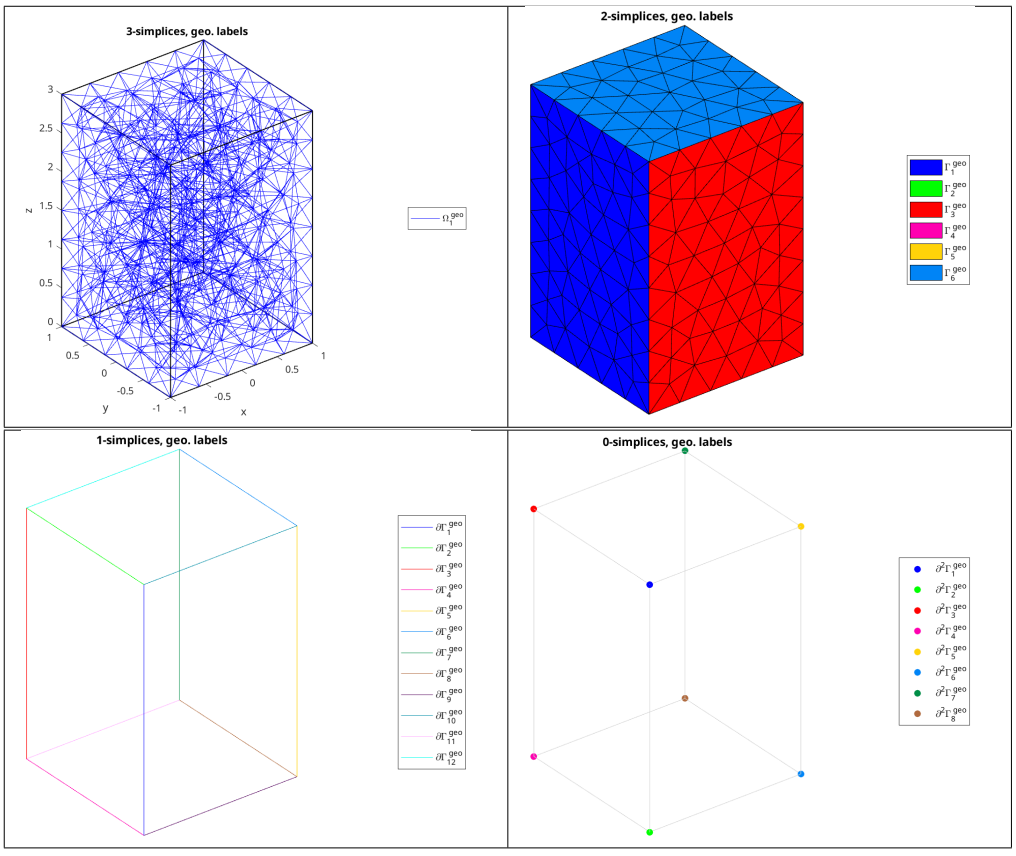


Figure 8: siMesh object corresponding to the box with parameters X = −1, Y = −1, Z = 0, DX = 2, DY = 2, DZ = 3. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.box(20,'d',2,'X',-1,'Y',-1,'Z',-1,'DX',2,'DY',2,'DZ',2);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 9 and in Figure 10.

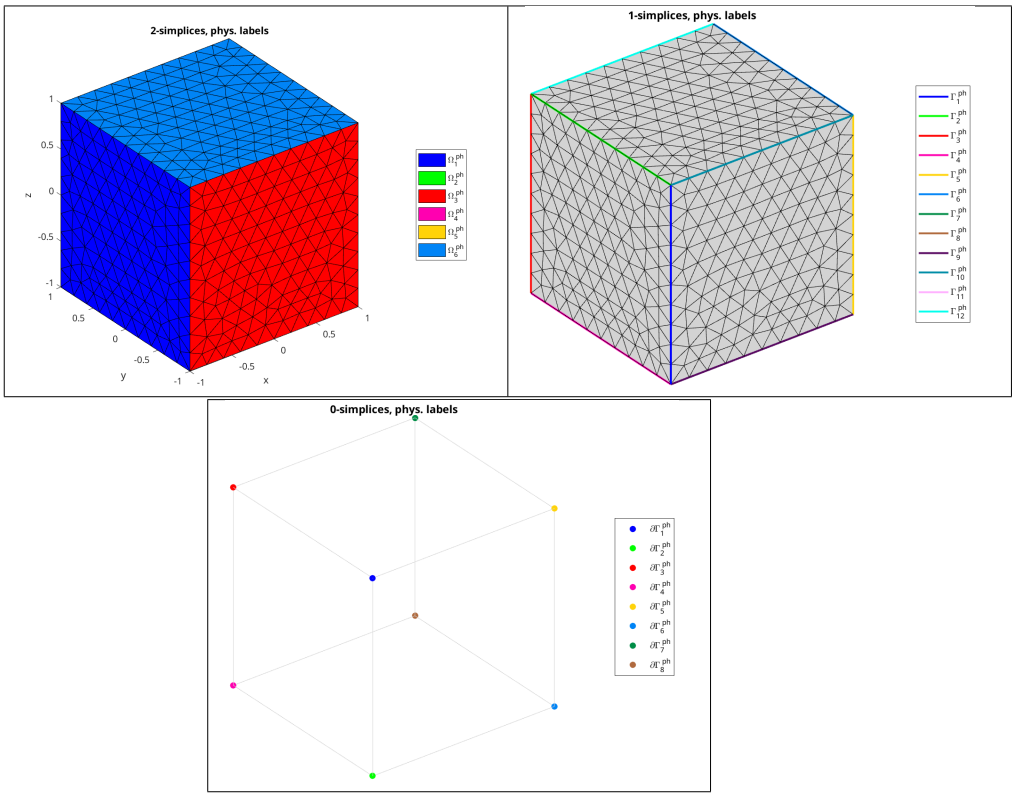


Figure 9: siMesh object corresponding to the surface of the box with parameters d = 2, X = −1, Y = −1, Z = −1, DX = 2, DY = 2, DZ = 2. Representation of the physical labels.

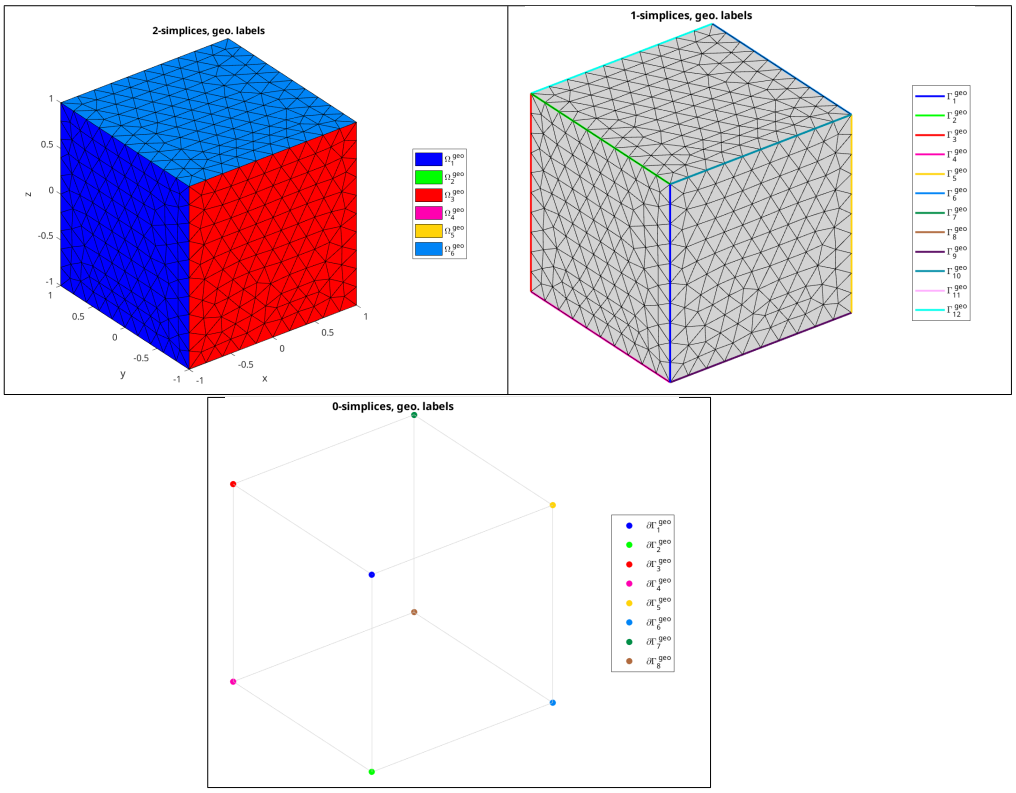


Figure 10: siMesh object corresponding to the surface of the box with parameters d = 2, X = −1, Y = −1, Z = −1, DX = 2, DY = 2, DZ = 2. Representation of the geometrical labels.

4.2.2 fc_simesh.samples.sphere function

The **fc_simesh**.samples.**sphere** function returns an siMesh object corresponding to a sphere obtain by using `gmsh` with `sphereOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.sphere(N)
Th=fc_simesh.samples.sphere(N,Name,Value, ...)
```

Description

fc_simesh.samples.**sphere**(N) returns a sphere as a siMesh object where N is a refinement parameter. By default, the sphere is centered in (X = 0, Y = 0, Z = 0) with radius R = 1.

fc_simesh.samples.**sphere**(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, d ∈ {2, 3}, volume mesh (3 default) or surface mesh (2),
- 'X' : to specify X value (default 0),
- 'Y' : to specify Y value (default 0),
- 'Z' : to specify Z value (default 0),
- 'R' : to specify R value (default 1)
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.sphere(20);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 11 and in Figure 12.

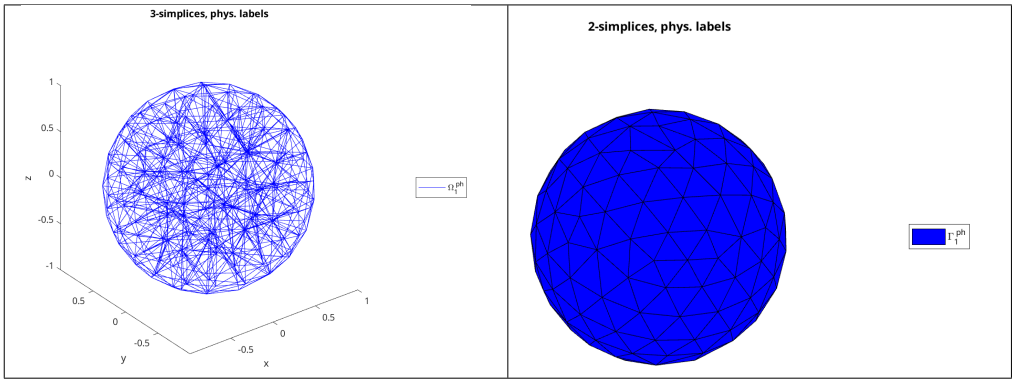


Figure 11: siMesh object corresponding to the default sphere (i.e. the unit sphere). Representation of the physical labels.

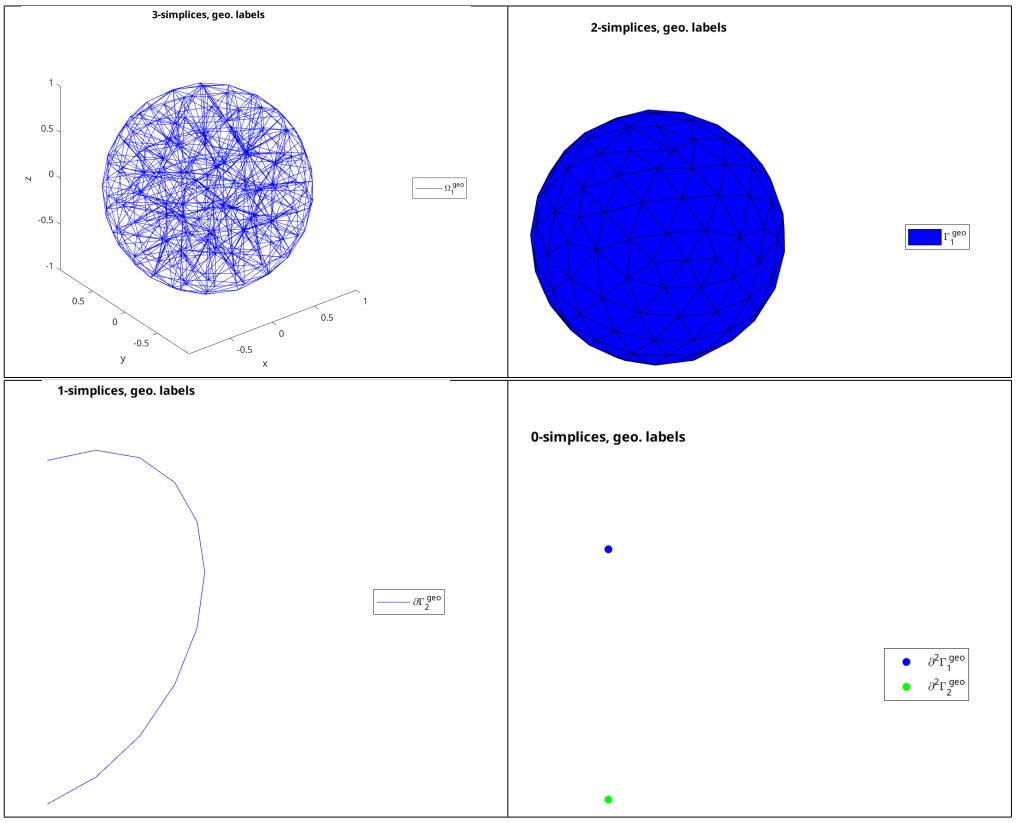


Figure 12: siMesh object corresponding to the default sphere (i.e. the unit sphere). Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.sphere(20,'X',-1,'Y',2,'Z',0,'R',2);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 13 and in Figure 14.

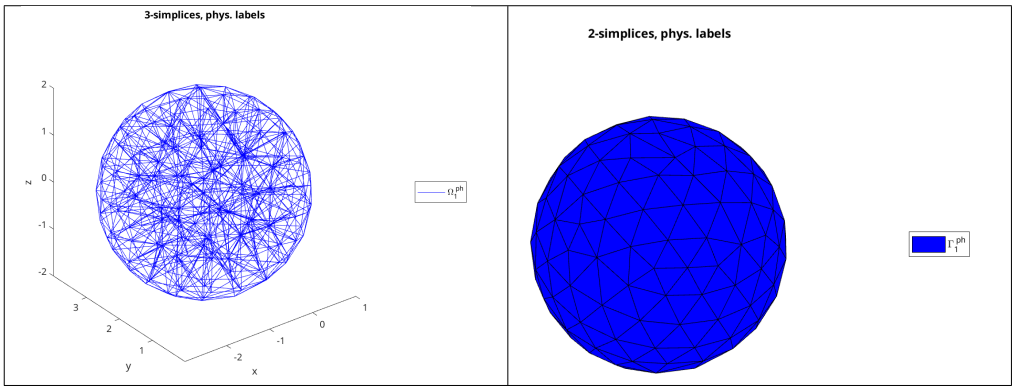


Figure 13: siMesh object corresponding to the sphere centered in $(-1, 2, 0)$ with radius R $= 2$. Representation of the physical labels.

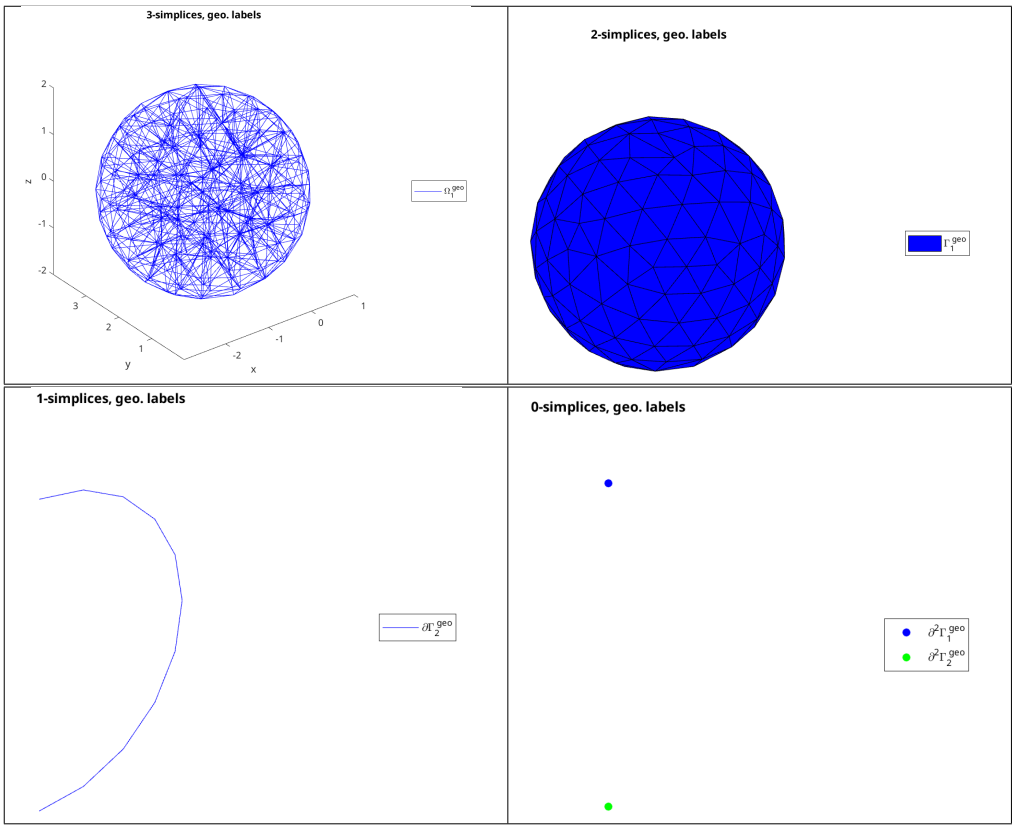


Figure 14: siMesh object corresponding to the sphere centered in $(-1, 2, 0)$ with radius R $= 2$. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.sphere(20,'d',2,'R',0.5);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 15 and in Figure 16.

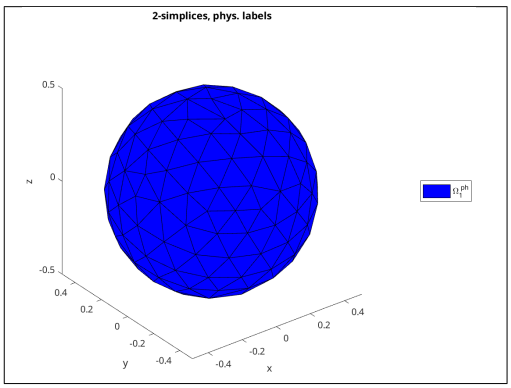


Figure 15: siMesh object corresponding to the surface of the sphere centered in $(0, 0, 0)$ with parameters d $= 2$, R $= 0.5$. Representation of the physical labels.

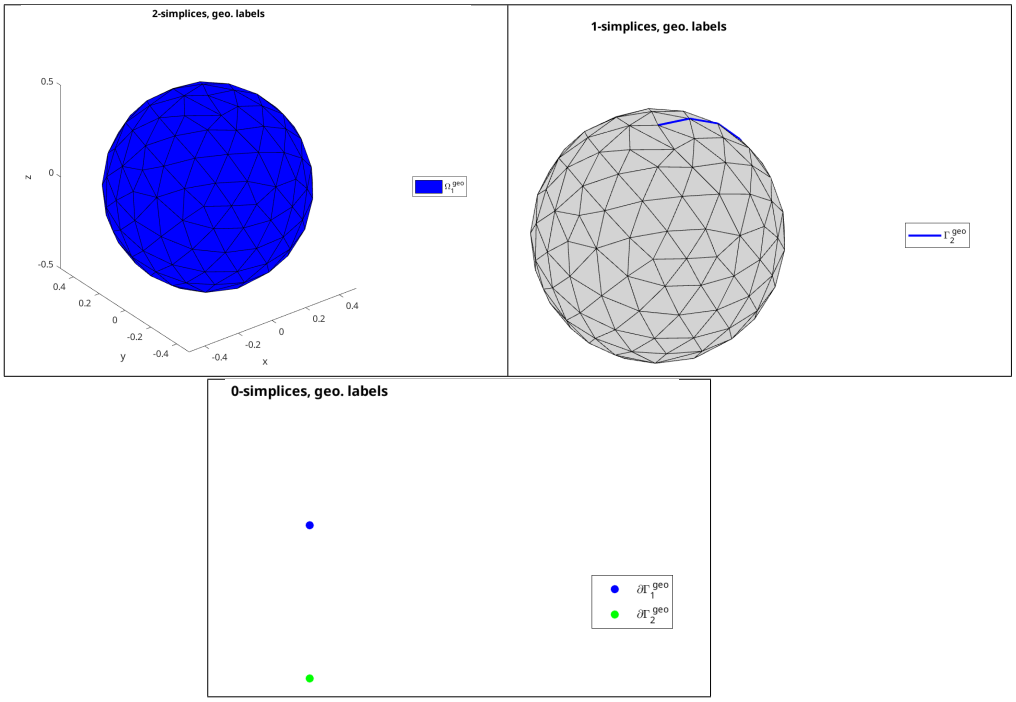


Figure 16: siMesh object corresponding to the surface of the sphere centered in $(0, 0, 0)$ with parameters d $= 2$, R $= 0.5$. Representation of the geometrical labels.

4.2.3 fc_simesh.samples.ellipsoid function

The **fc_simesh**.samples.ellipsoid function returns an siMesh object corresponding to a ellipsoid obtain by using `gmsh` with `ellipsoidOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.ellipsoid(N)
Th=fc_simesh.samples.ellipsoid(N,Name,Value, ...)
```

Description

fc_simesh.samples.ellipsoid(N) returns a ellipsoid as a siMesh object where N is a refinement parameter. By default, the ellipsoid is centered in (X $= 0$, Y $= 0$, Z $= 0$) with radius RX $= 1$, RY $= 1$ and RZ $= 1$.

fc_simesh.samples.ellipsoid(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, d $\in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),

- 'X' : to specify X value (default 0),
- 'Y' : to specify Y value (default 0),
- 'Z' : to specify Z value (default 0),
- 'RX' : to specify RX value (default 1), radius along x-axis,
- 'RY' : to specify RY value (default 1), radius along y-axis,
- 'RZ' : to specify RZ value (default 1), radius along z-axis,
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.ellipsoid(15)
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 17 and in Figure 18.

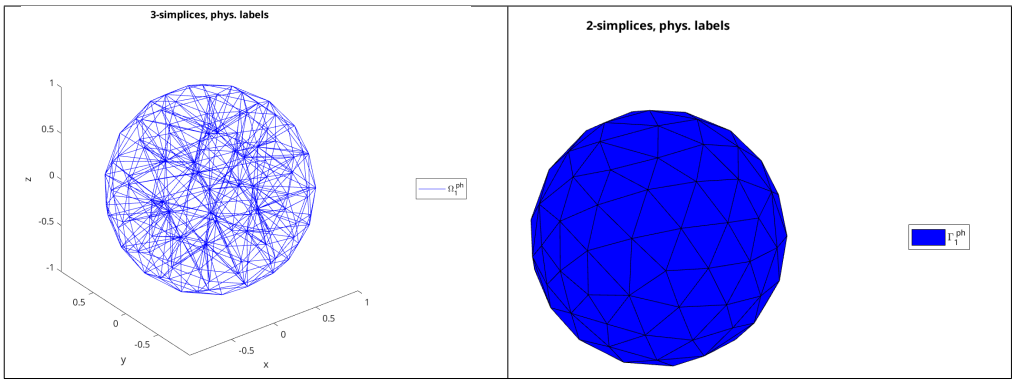


Figure 17: siMesh object corresponding to the default ellipsoid (i.e. the unit sphere). Representation of the physical labels.

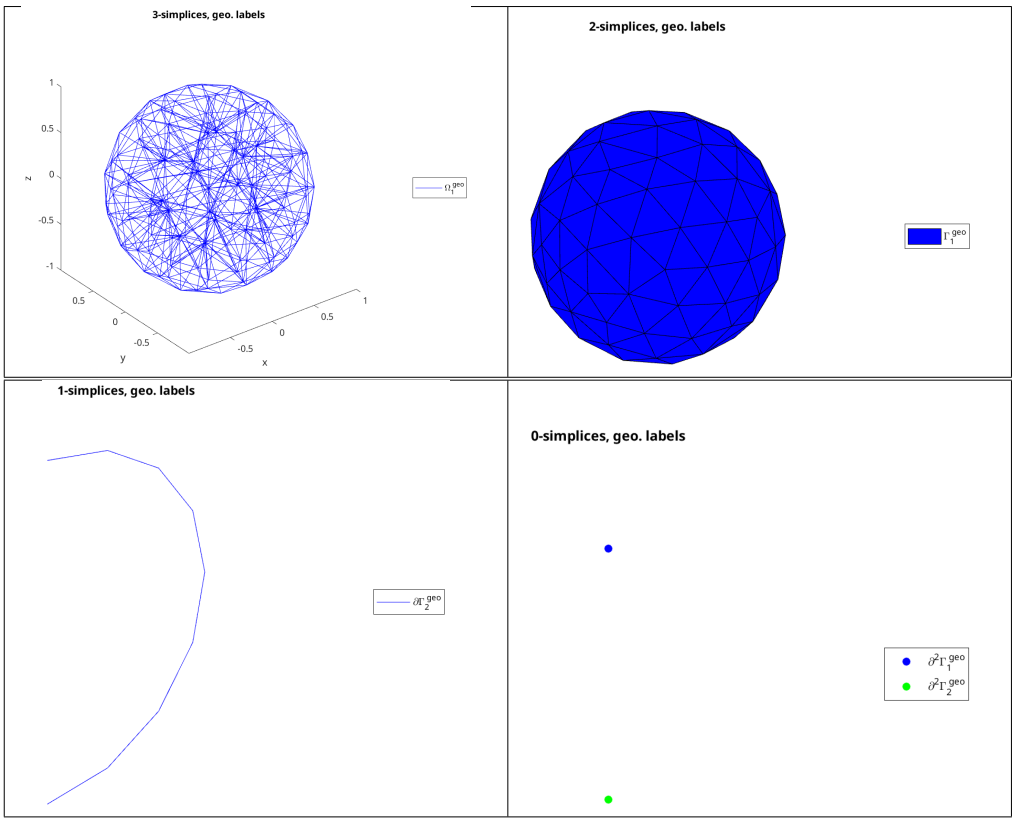


Figure 18: siMesh object corresponding to the default ellipsoid (i.e. the unit sphere). Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.ellipsoid(15,'X',-1,'Y',2,'Z',0,'RX',2,'RZ',0.5);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 19 and in Figure 20.

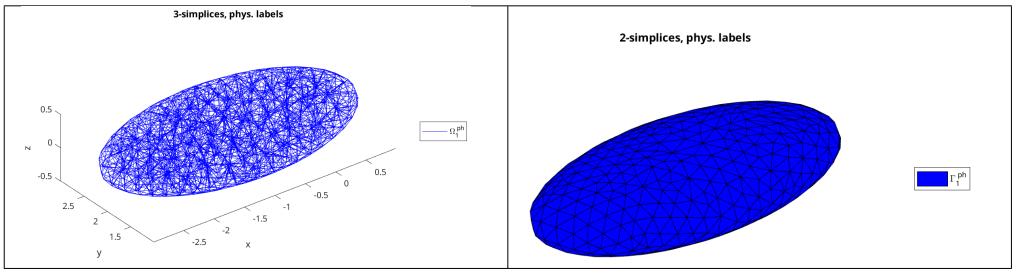


Figure 19: siMesh object corresponding to the ellipsoid centered in $(-1, 2, 0)$ with radii RX $= 2$, RY $= 1$ and RZ $= 1$. Representation of the physical labels.

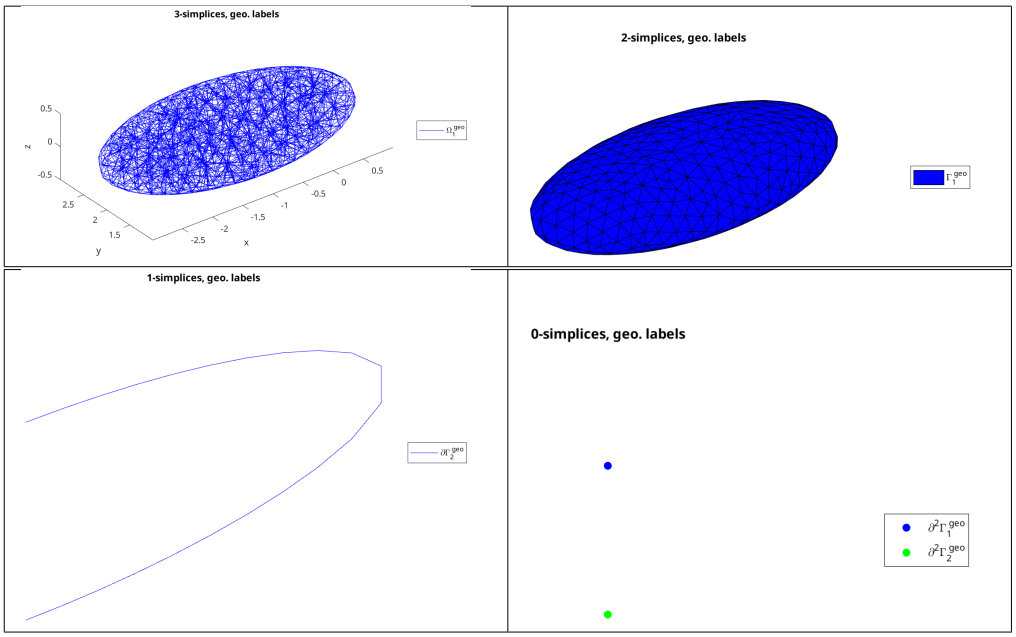


Figure 20: siMesh object corresponding to the ellipsoid centered in $(-1, 2, 0)$ with radii RX $= 2$, RY $= 1$ and RZ $= 1$. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.ellipsoid(15,'d',2,'RX',1.5,'RZ',0.5);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 21 and in Figure 22.

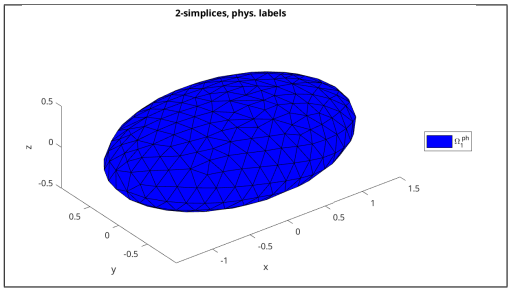


Figure 21: siMesh object corresponding to the surface of the ellipsoid centered in $(0, 0, 0)$ with parameters d $= 2$, RX $= 1.5$, RY $= 1$ and RZ $= 0.5$. Representation of the physical labels.

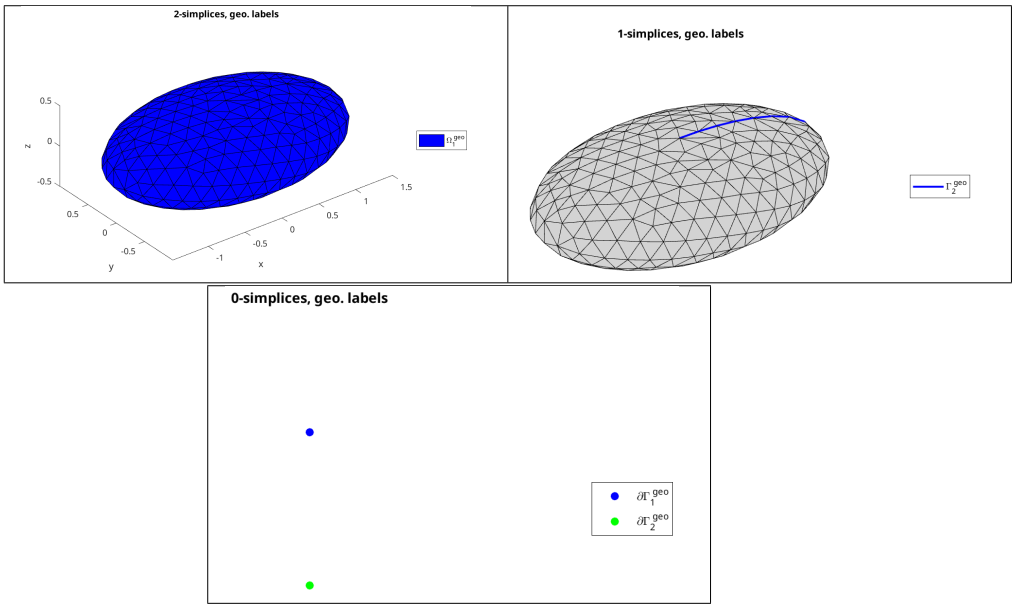


Figure 22: siMesh object corresponding to the surface of the ellipsoid centered in $(0, 0, 0)$ with parameters d $= 2$, RX $= 1.5$, RY $= 1$ and RZ $= 0.5$. Representation of the geometrical labels.

4.2.4 fc_simesh.samples.cylinder function

The **fc_simesh**.samples.**cylinder** function returns an siMesh object corresponding to a cylinder obtain by using `gmsh` with `cylinderOC.geo` file. The cylinder is defined by

- the 3 coordinates of the center of the first circular face (X, (Y, (Z), default $(0, 0, 0)$,
- the 3 components of the vector defining its axis (DX, (DY, (DZ), default $(0, 0, 1)$,
- its radius R, default 0.5,
- the angular opening theta $\in]0, 2 * \pi]$, default 2π.

Syntaxe

```
Th=fc_simesh.samples.cylinder(N)
Th=fc_simesh.samples.cylinder(N,Name,Value, ...)
```

Description

fc_simesh.samples.**cylinder**(N) returns the default cylinder as a siMesh object where N is a refinement parameter.

fc_simesh.samples.**cylinder**(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, d $\in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'X' : to specify X value (default 0),
- 'Y' : to specify Y value (default 0),
- 'Z' : to specify Z value (default 0),
- 'DX' : to specify DX value (default 0),
- 'DY' : to specify DY value (default 0),
- 'DZ' : to specify DZ value (default 1),
- 'R' : to specify R value (default 0.5),
- 'theta' : to specify theta value (default 2π)
- 'verbose' : to specify verbosity from 0 to 4... (default 0, no verbosity)
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.cylinder(20)
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 23 and in Figure 24.

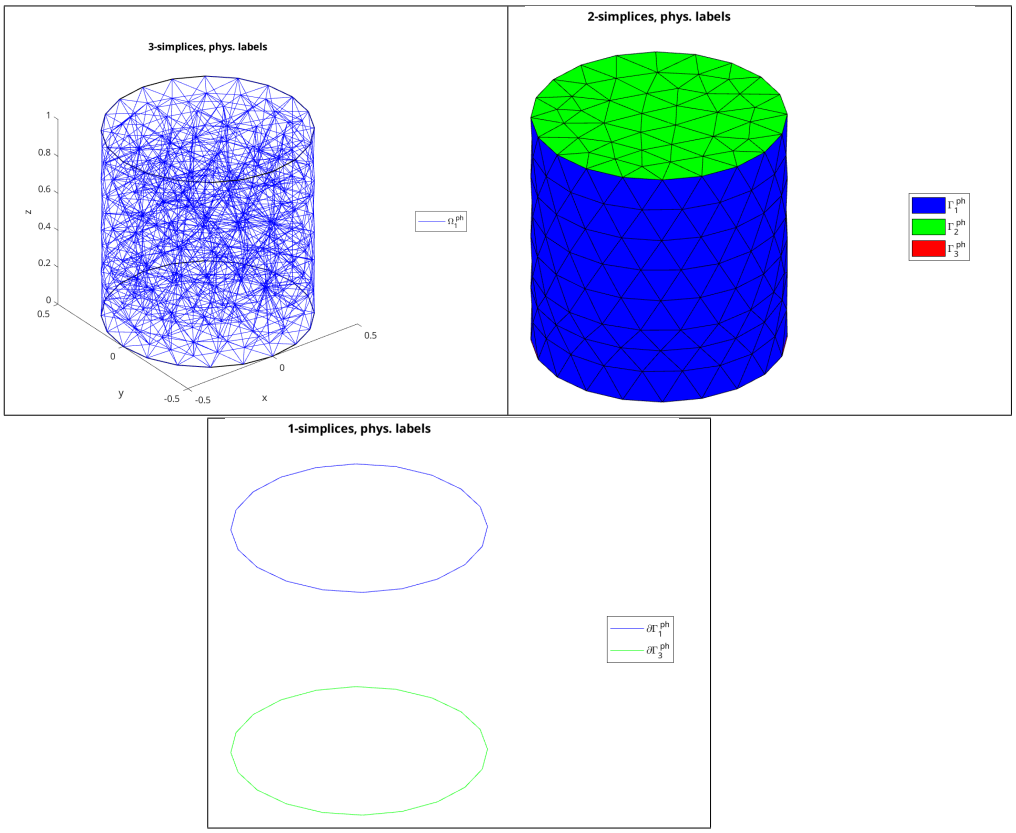


Figure 23: siMesh object corresponding to the default cylinder. Representation of the physical labels.

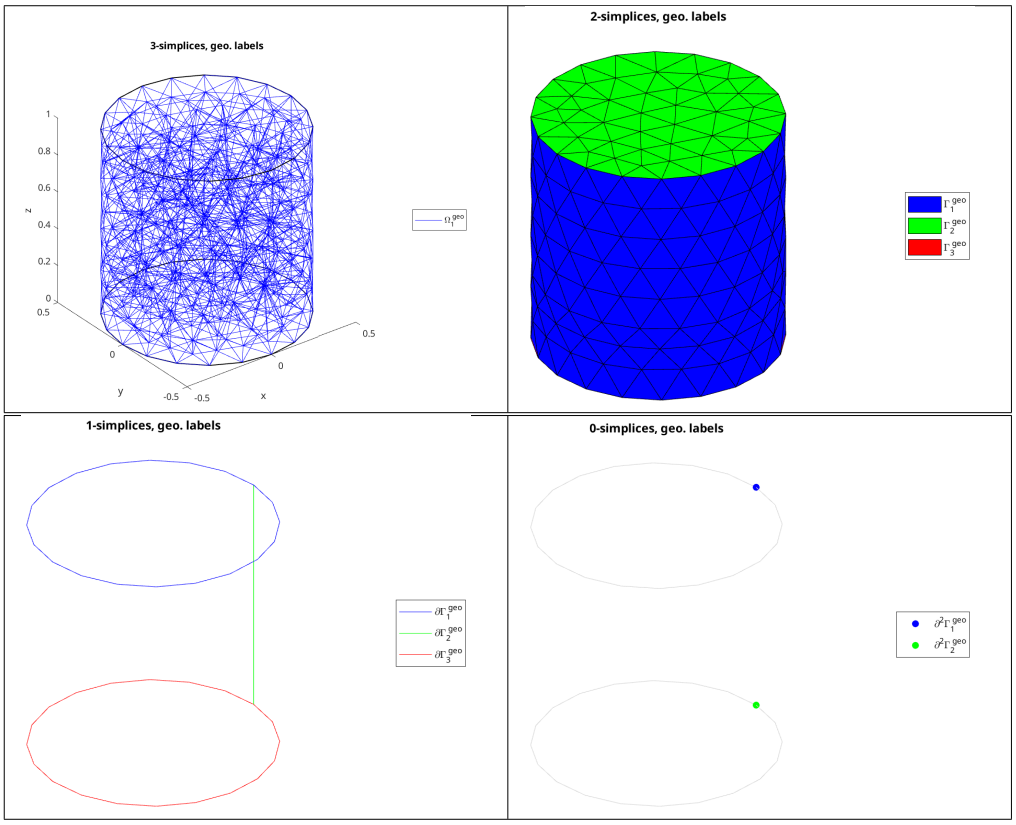


Figure 24: siMesh object corresponding to the default cylinder. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.cylinder(25,'X',-1,'Y',2,'Z',0,'DX',0,'DY',3,'DZ',0,'R',1,'theta',3*pi/2)
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 25 and in Figure 26.

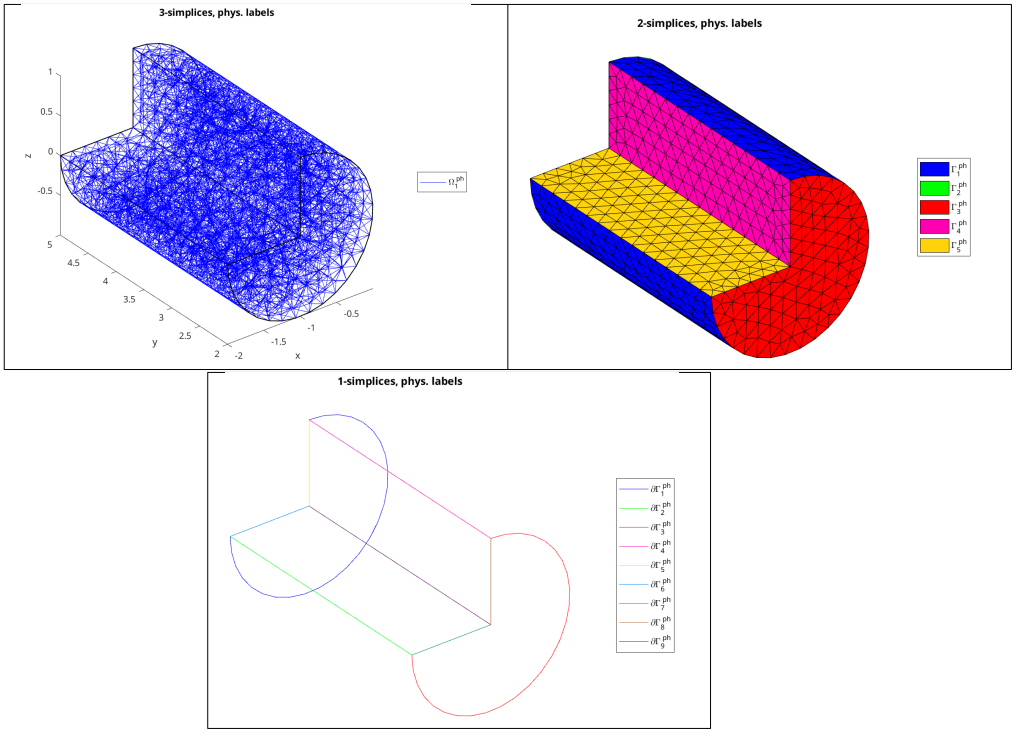


Figure 25: siMesh object corresponding to the cylinder with parameter X $= -1$, Y $= 2$, Z $= 0$, DX $= 0$, DY $= 3$, DZ $= 0$, R $= 1$ and theta $= \frac{3\pi}{2}$. Representation of the physical labels.

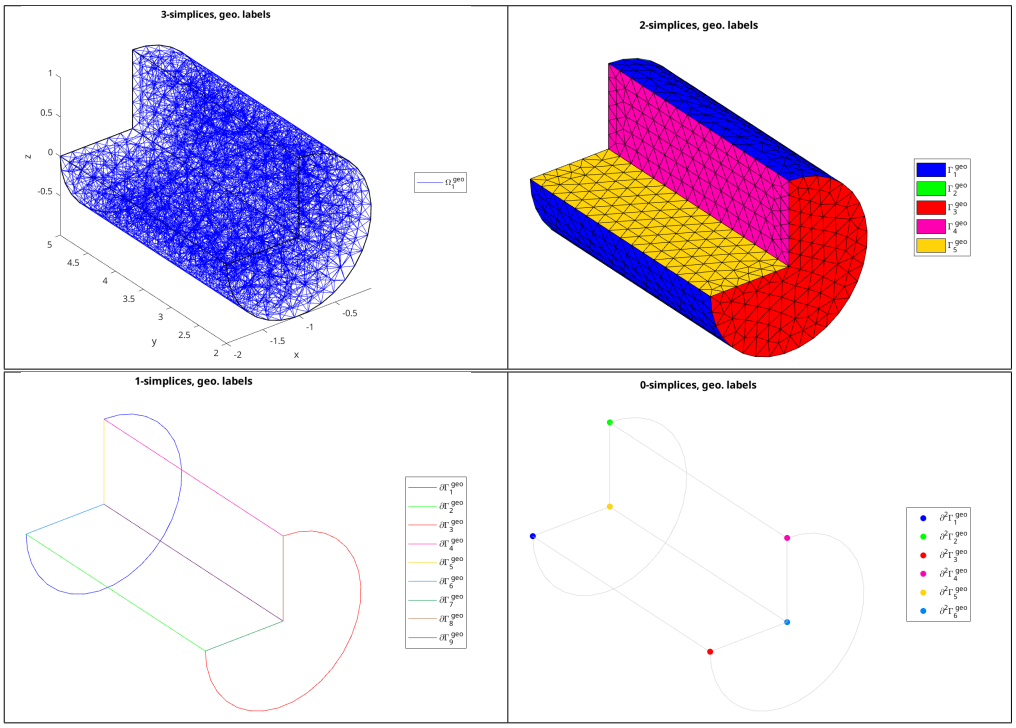


Figure 26: siMesh object corresponding to the cylinder with parameter X = −1, Y = 2, Z = 0, DX = 0, DY = 3, DZ = 0, R = 1 and theta = $\frac{3\pi}{2}$. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.cylinder(25,'d',2,'DX',1,'DY',1,'DZ',1,'R',0.5,'theta',3*pi/4)
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 27 and in Figure 28.

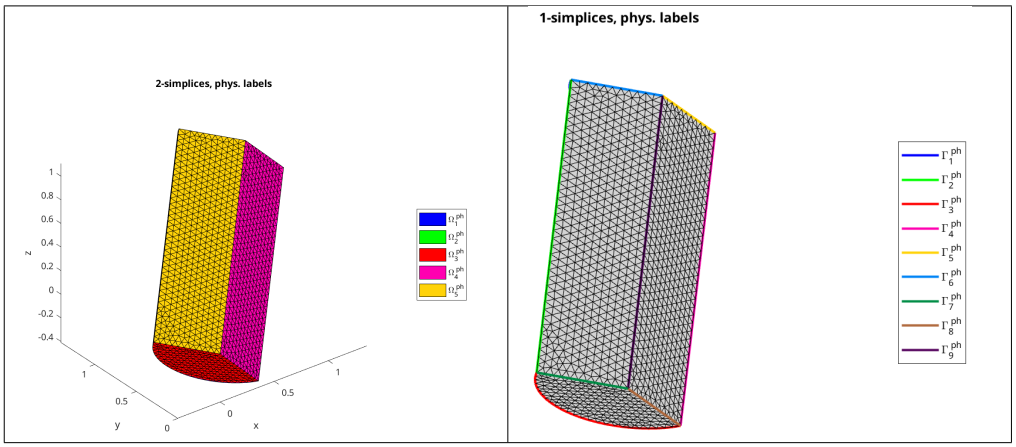


Figure 27: siMesh object corresponding to the surface of the cylinder with parameter X = 0, Y = 0, Z = 0, DX = 1, DY = 1, DZ = 1, R = 0.5 and theta = $\frac{5\pi}{4}$.. Representation of the physical labels.

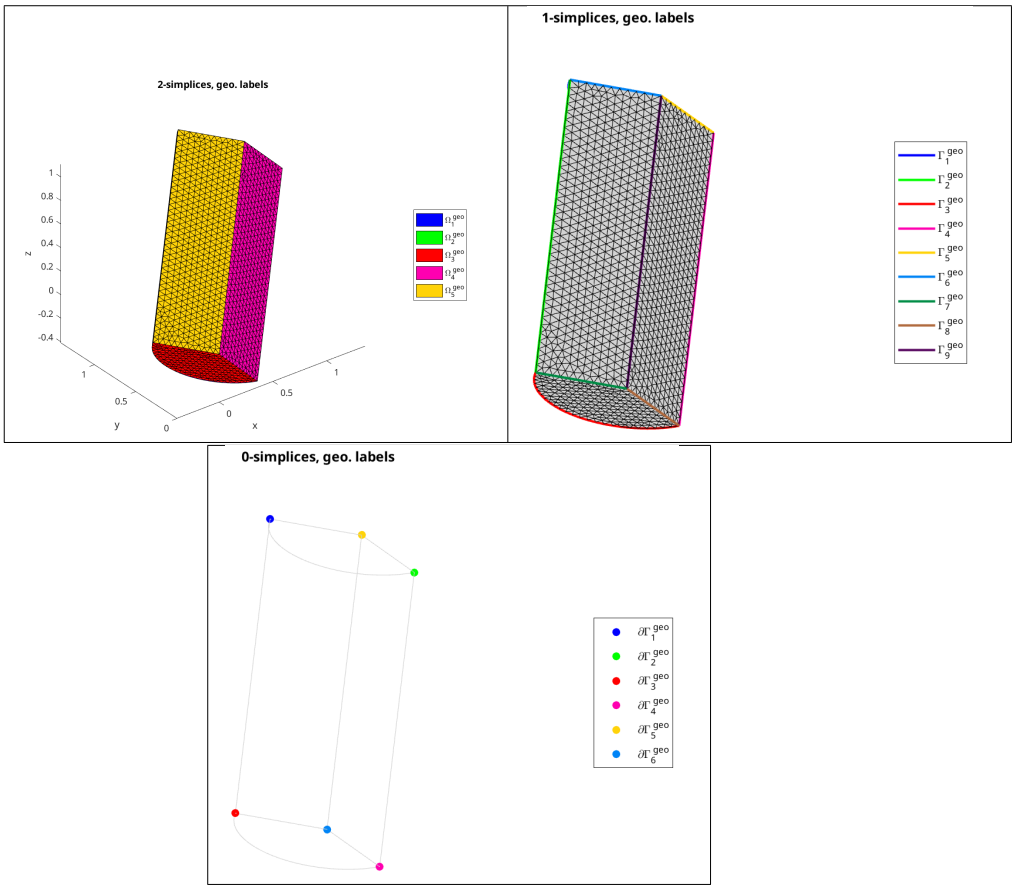


Figure 28: siMesh object corresponding to the surface of the cylinder with parameter X = 0, Y = 0, Z = 0, DX = 1, DY = 1, DZ = 1, R = 0.5 and theta = $\frac{5\pi}{4}$.. Representation of the geometrical labels.

4.2.5 fc_simesh.samples.cone function

The **fc_simesh**.samples.cone function returns an siMesh object corresponding to a cone obtain by using `gmsh` with `coneOC.geo` file. The cone is defined by

- the 3 coordinates of the center of the first circular face (X, (Y, (Z), default $(0, 0, 0)$,
- the 3 components of the vector defining its axis (DX, (DY, (DZ), default $(0, 0, 1)$,
- two radii of the faces (these radii can be zero), R1, default 0.5, R2, default 0.2,
- the angular opening theta $\in]0, 2 * \pi]$, default 2π.

Syntaxe

```
Th=fc_simesh.samples.cone(N)
Th=fc_simesh.samples.cone(N,Name,Value, ...)
```

Description

fc_simesh.samples.cone(N) returns the default cone as a siMesh object where N is a refinement parameter.

fc_simesh.samples.cone(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, d $\in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'X' : to specify X value (default 0),
- 'Y' : to specify Y value (default 0),
- 'Z' : to specify Z value (default 0),

- 'DX' : to specify DX value (default 0),
- 'DY' : to specify DY value (default 0),
- 'DZ' : to specify DZ value (default 1),
- 'R1' : to specify R1 value (default 0.5)
- 'R2' : to specify R2 value (default 0.2)
- 'theta' : to specify theta value (default 2π)
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.cone(30)
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 29 and in Figure 30.

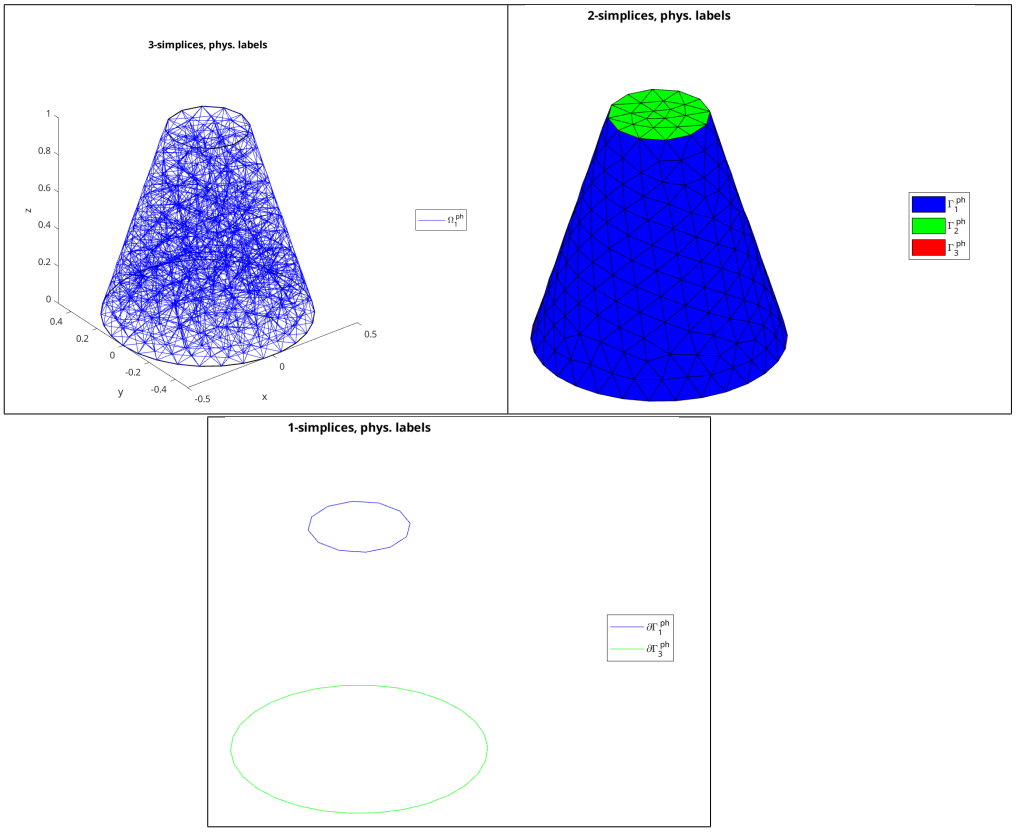


Figure 29: siMesh object corresponding to the default cone. Representation of the physical labels.

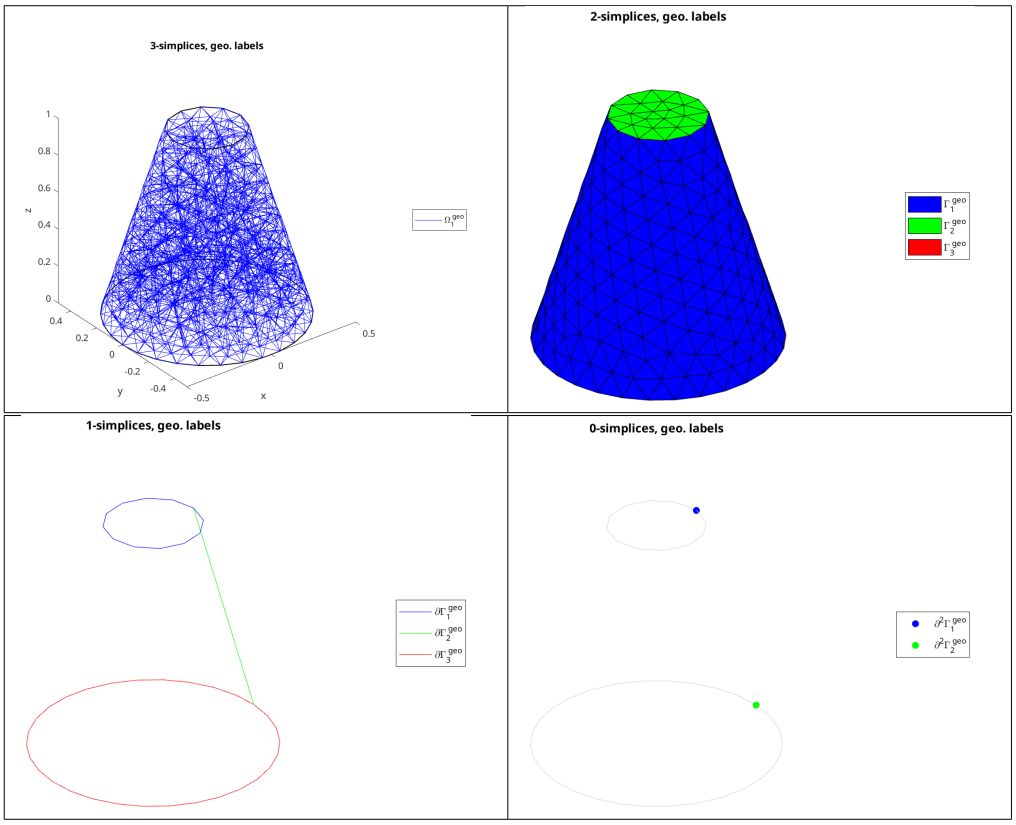


Figure 30: siMesh object corresponding to the default cone. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.cone(30,'X',-1,'Y',2,'Z',0,'DX',0,'DY',3,'DZ',0,'R1',0.3,'R2',0.8,'theta',3*
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 31 and in Figure 32.

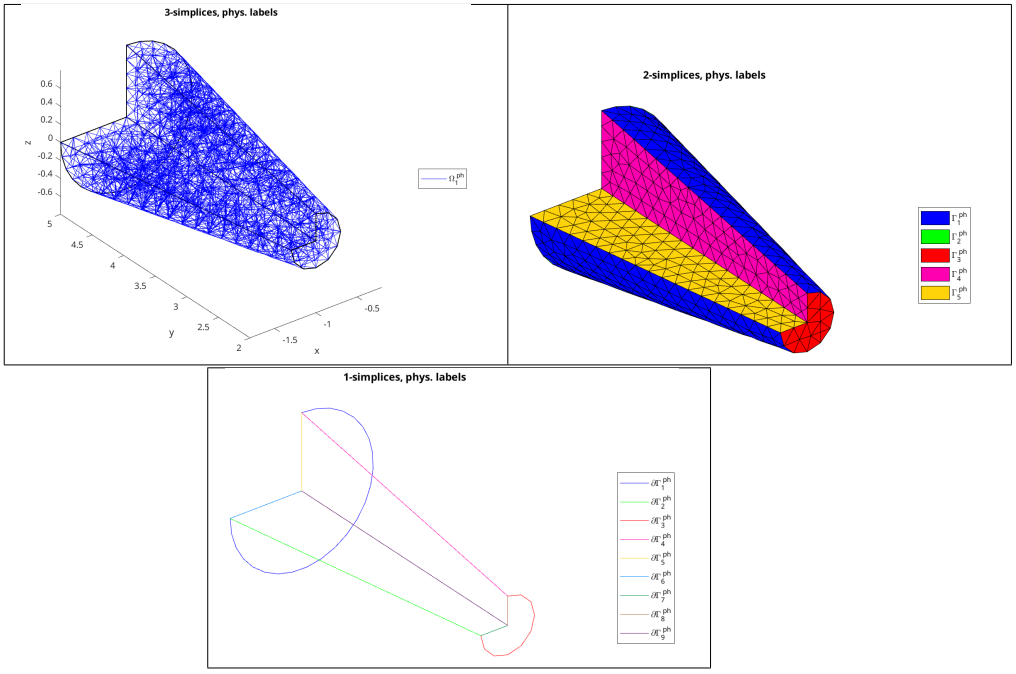


Figure 31: siMesh object corresponding to the cone with parameters X $= -1$, Y $= 2$, Z $= 0$, DX $= 0$, DY $= 3$, DZ $= 0$, R $= 1$ and theta $= \frac{3\pi}{2}$. Representation of the physical labels.

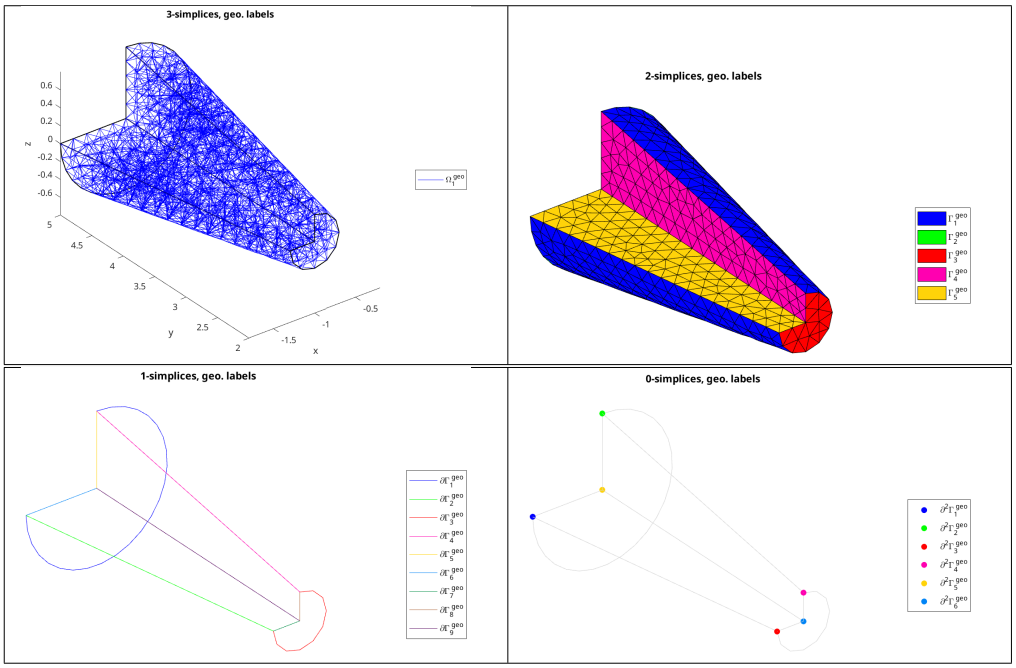


Figure 32: siMesh object corresponding to the cone with parameters X = −1, Y = 2, Z = 0, DX = 0, DY = 3, DZ = 0, R = 1 and theta = $\frac{3\pi}{2}$. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.cone(30,'d',2,'DX',1,'DY',0.5,'DZ',1,'R1',0.5,'R2',0,'theta',3*pi/4)
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 33 and in Figure 34.

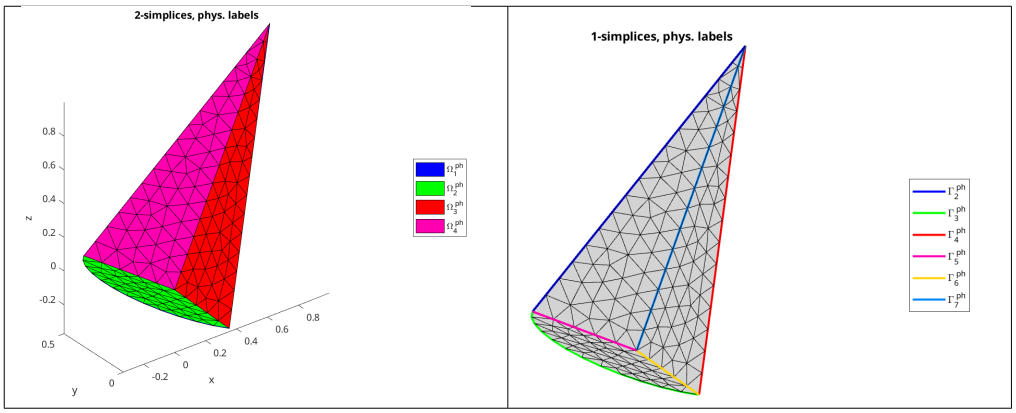


Figure 33: siMesh object corresponding to the surface of the cone with parameters d = 2, X = 0, Y = 0, Z = 0, DX = 1, DY = 0.5, DZ = 1, R1 = 0.5, R2 = 0, and theta = $\frac{3\pi}{4}$.. Representation of the physical labels.

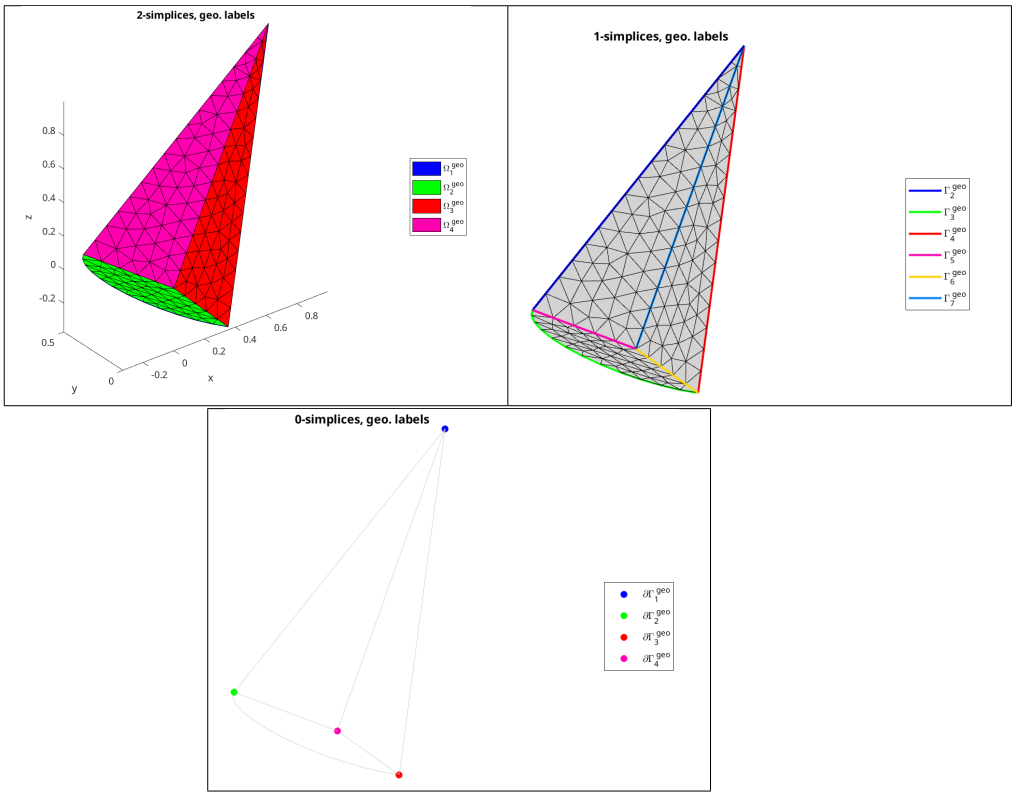


Figure 34: siMesh object corresponding to the surface of the cone with parameters d = 2, X = 0, Y = 0, Z = 0, DX = 1, DY = 0.5, DZ = 1, R1 = 0.5, R2 = 0, and theta = $\frac{3\pi}{4}$.. Representation of the geometrical labels.

4.2.6 fc_simesh.samples.dice function

The **fc_simesh**.samples.dice function returns an siMesh object corresponding to a dice obtain by using `gmsh` with `diceOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.dice(N)
Th=fc_simesh.samples.dice(N,Name,Value, ...)
```

Description

fc_simesh.samples.dice(N) returns a dice as a siMesh object where N is a refinement parameter. By default, the dice is centered in $(0,0,0)$ and its the intersection between the box $[-L/2, L/2]^3$, L = 1 by default, and the sphere centered in $(0,0,0)$ with radius $R = -\frac{L}{2} + T\left(\frac{L}{2}(\sqrt{2}-1)\right)$ with T $\in]0,1[$ (default 0.7).

fc_simesh.samples.dice(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, d $\in \{2,3\}$, volume mesh (3 default) or surface mesh (2),
- 'L' : to specify L value (default 1),
- 'T' : to specify T value (default 0.7),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.dice(15)
```

The physical and geometrical labels of the **fc_simesh**.siMesh object Th are respectively represented in Figure 35 and in Figure 36.

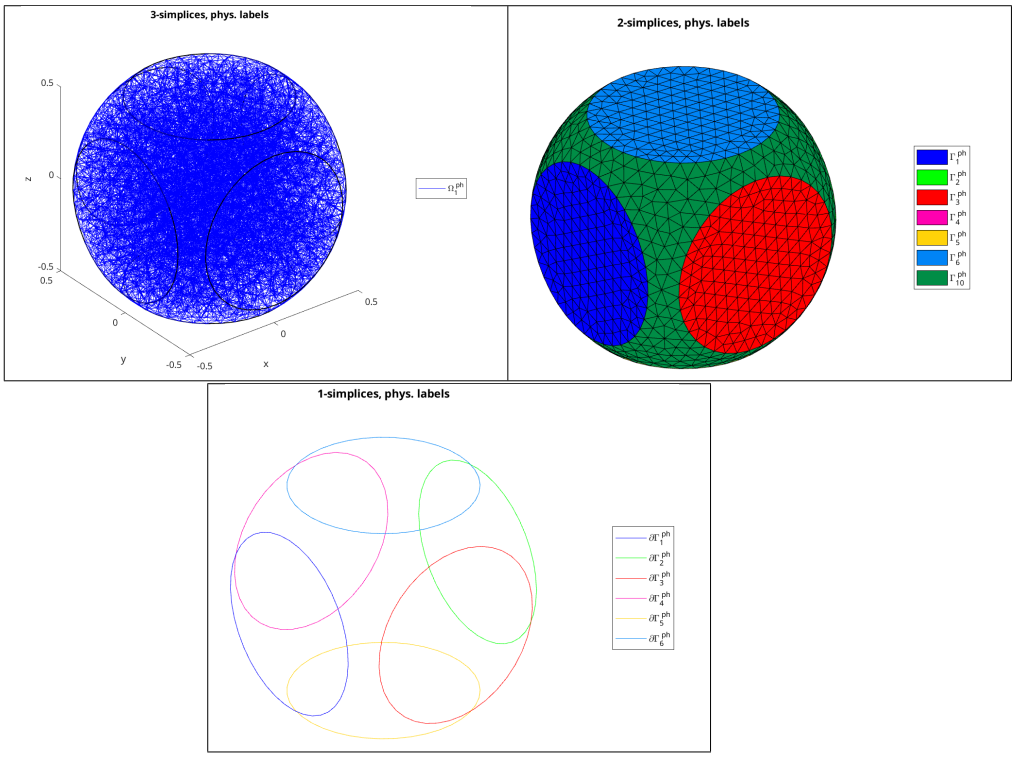


Figure 35: siMesh object corresponding to the default dice. Representation of the physical labels.

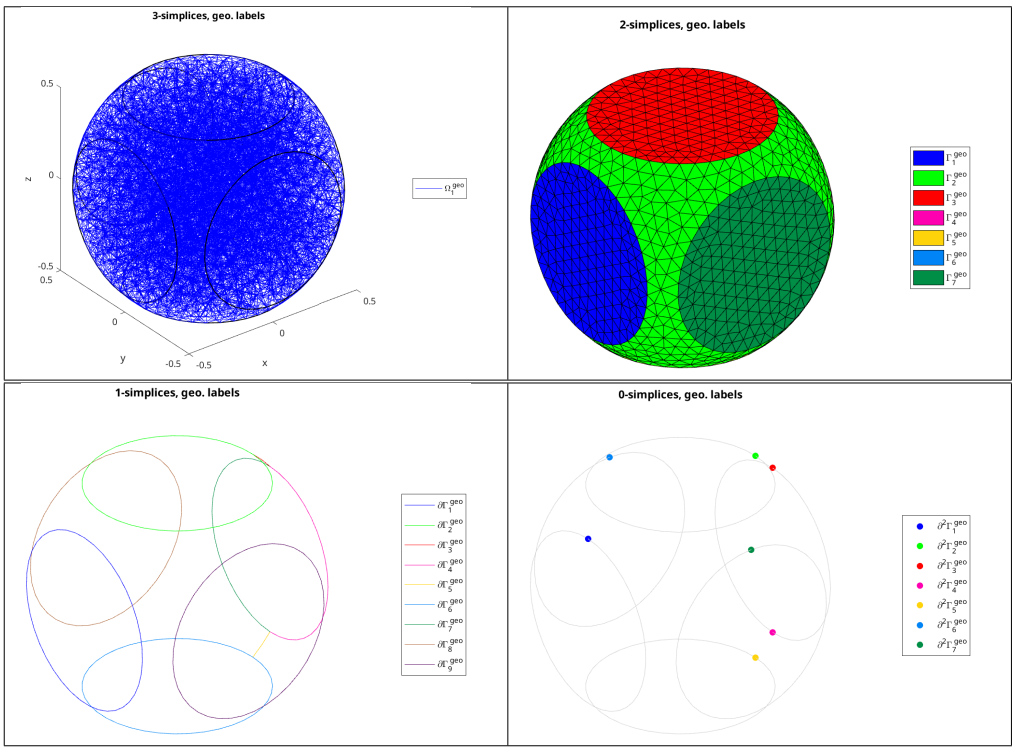


Figure 36: siMesh object corresponding to the default dice. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.dice(15,'L',2,'T',0.2);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 37 and in Figure 38.

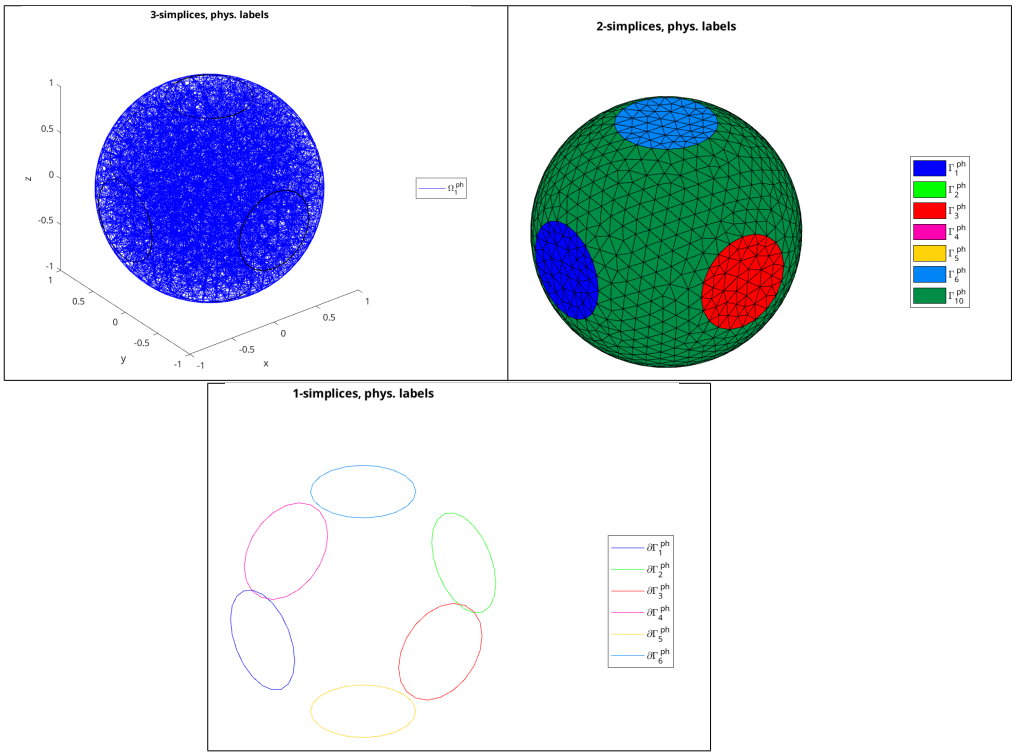


Figure 37: siMesh object corresponding to the dice with L = 2 and T = 0.2. Representation of the physical labels.

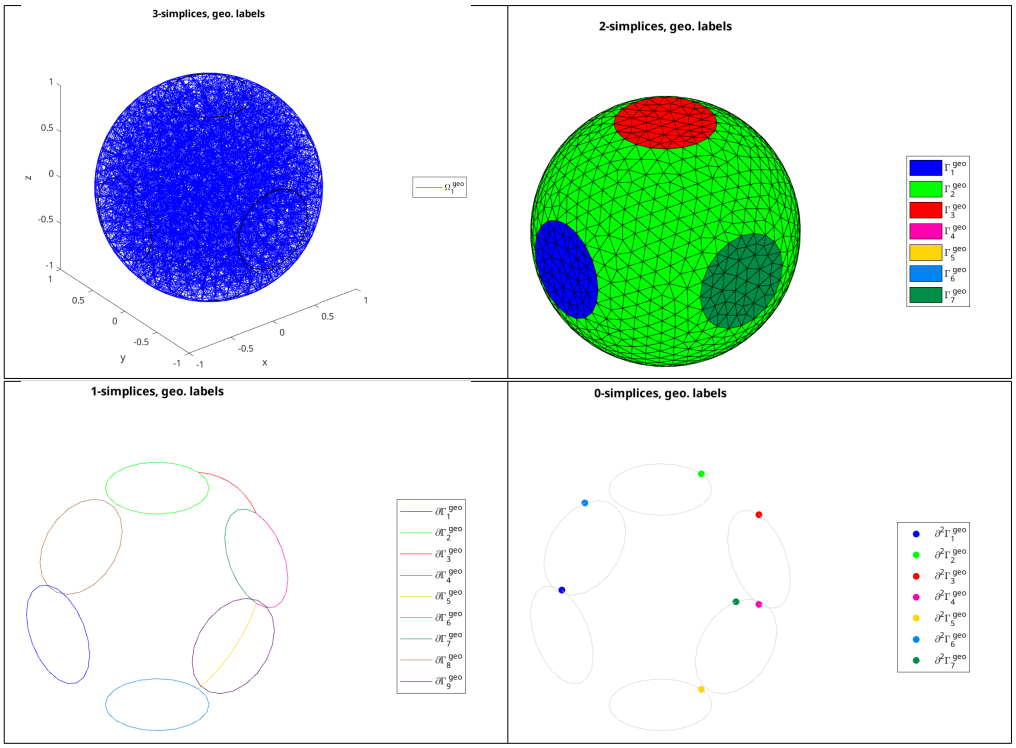


Figure 38: siMesh object corresponding to the dice with L = 2 and T = 0.2. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.dice(15,'d',2,'T',0.85);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 39 and in Figure 40.

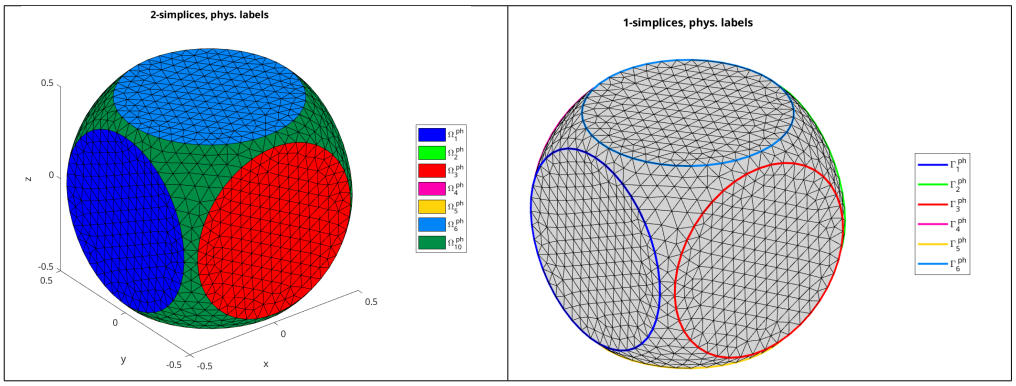


Figure 39: siMesh object corresponding to the surface of the dice with parameters L = 1 and T = 0.85. Representation of the physical labels.

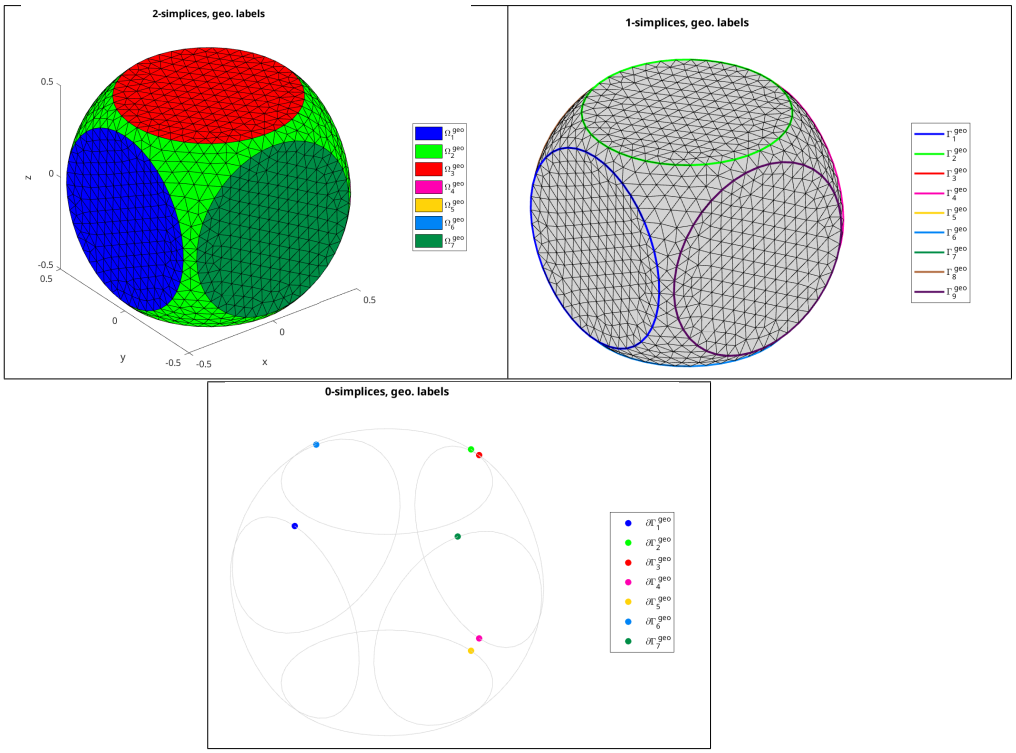


Figure 40: siMesh object corresponding to the surface of the dice with parameters L = 1 and T = 0.85. Representation of the geometrical labels.

4.2.7 fc_simesh.samples.wedge function

The **fc_simesh**.samples.wedge function returns an siMesh object corresponding to a right angular wedge obtain by using `gmsh` with `wedgeOC.geo` file. The right angular wedge is defined by

- the 3 coordinates of the right-angle point (X, (Y, (Z), default $(0, 0, 0)$,
- the 3 extends (DX, (DY, (DZ), default $(1, 1, 1)$,
- the top X extent topDX, default 0.

Syntaxe

```
Th=fc_simesh.samples.wedge(N)
Th=fc_simesh.samples.wedge(N,Name,Value, ...)
```

Description

fc_simesh.samples.wedge(N) returns the default wedge as a siMesh object where N is a refinement parameter.

fc_simesh.samples.wedge(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, $d \in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'X' : to specify X value (default 0),
- 'Y' : to specify Y value (default 0),
- 'Z' : to specify Z value (default 0),
- 'DX' : to specify DX value (default 1),
- 'DY' : to specify DY value (default 1),
- 'DZ' : to specify DZ value (default 1),
- 'topDX' : to specify topDX value (default 0),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.wedge(10);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 41 and in Figure 42.

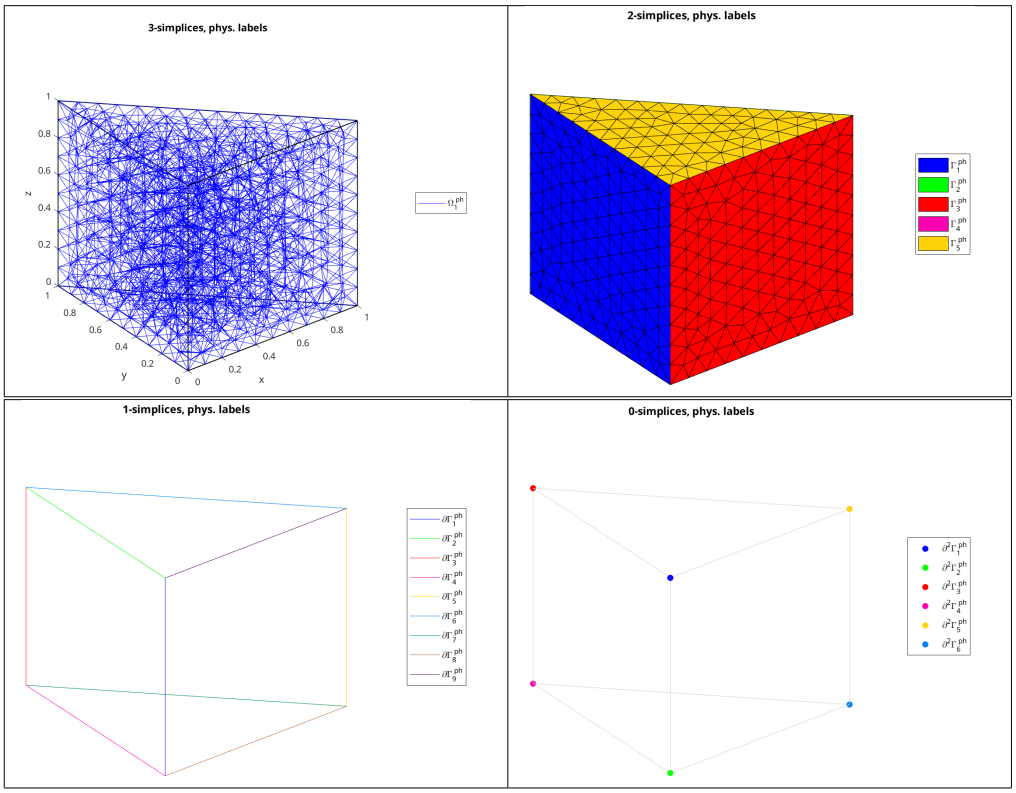


Figure 41: siMesh object corresponding to the default wedge. Representation of the physical labels.

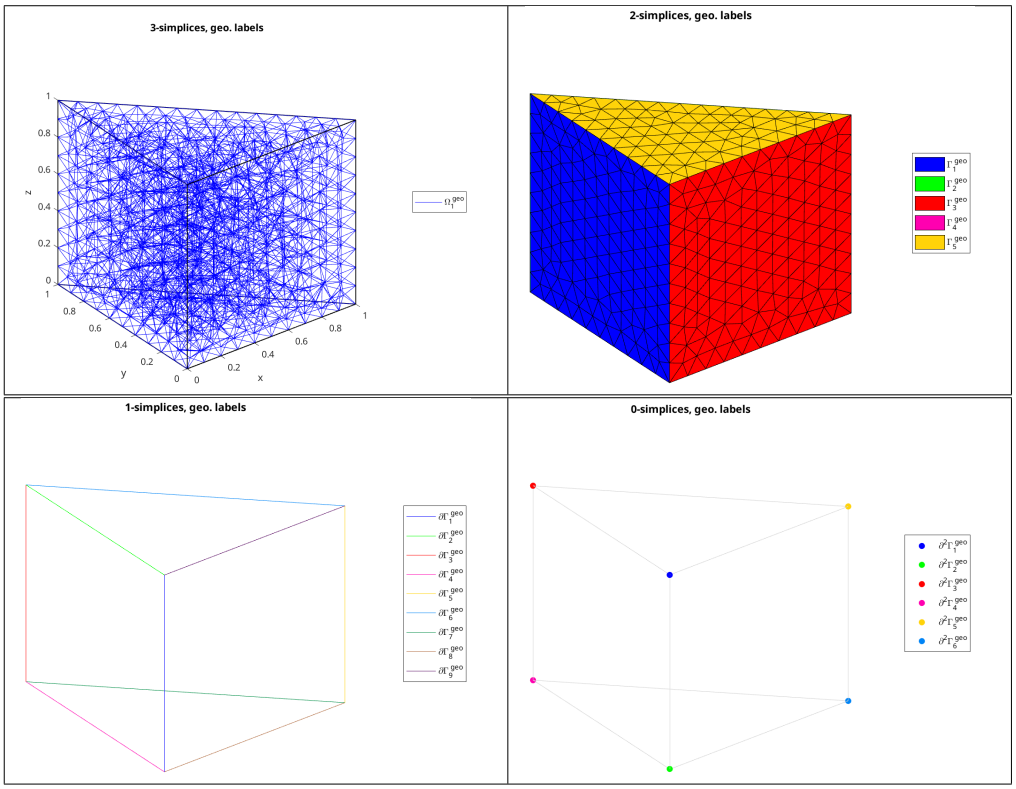


Figure 42: siMesh object corresponding to the default wedge. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.wedge(10,'X',-1,'Y',2,'Z',0,'DX',1,'DY',0.75,'DZ',1.5,'topDX',0.3);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 43 and in Figure 44.

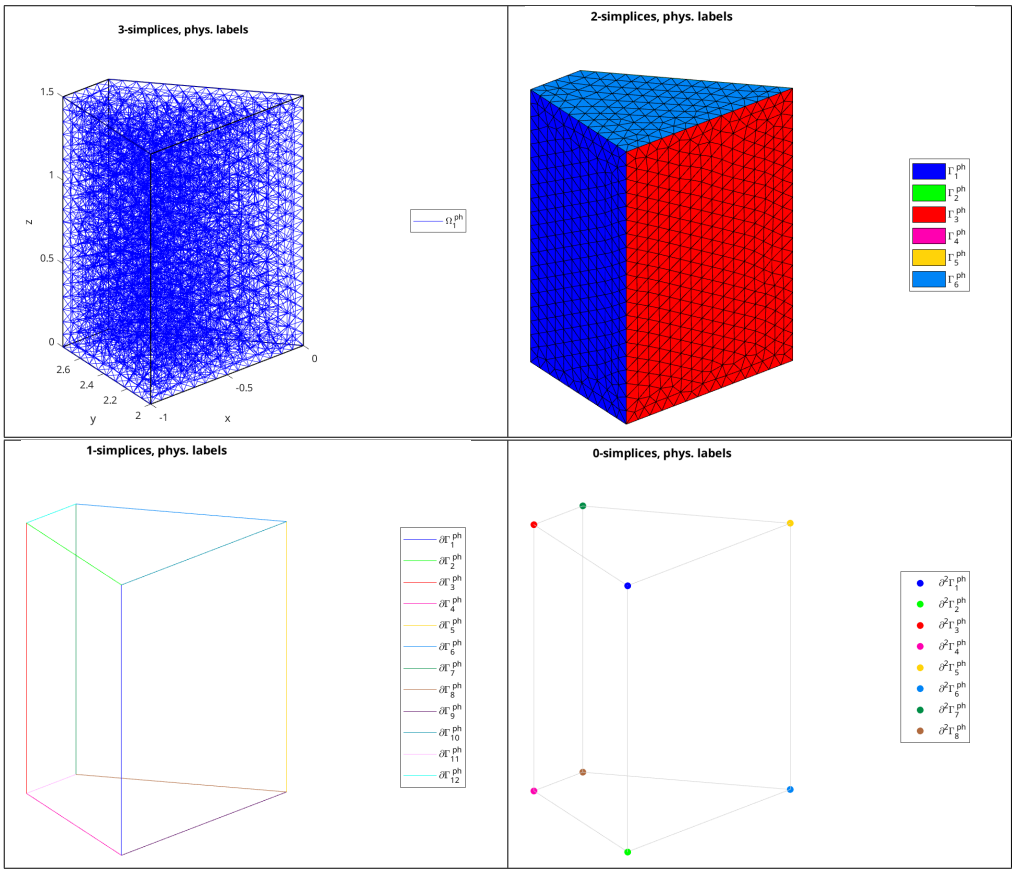


Figure 43: siMesh object corresponding to the wedge with parameters X = −1, Y = 2, Z = 0, DX = 1, DY = 0.75, DZ = 1.5, topDX = 0.3. Representation of the physical labels.

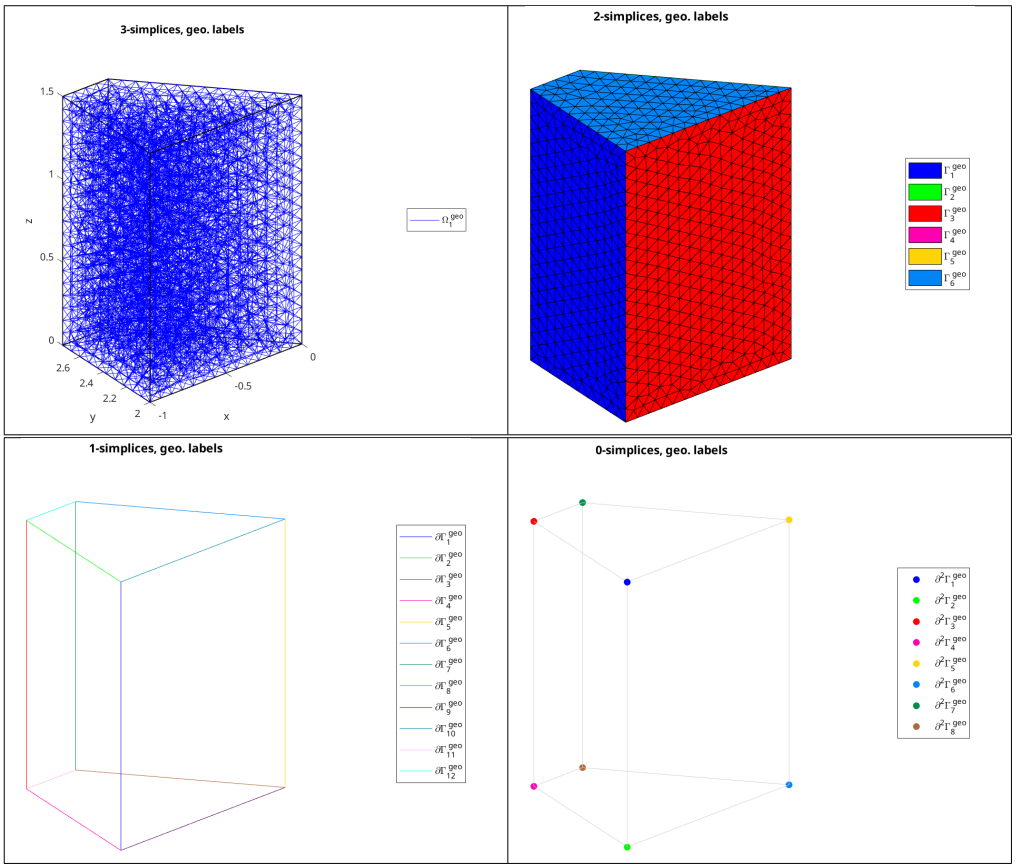


Figure 44: siMesh object corresponding to the wedge with parameters X = −1, Y = 2, Z = 0, DX = 1, DY = 0.75, DZ = 1.5, topDX = 0.3. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.wedge(10,'d',2,'topDX',0.3);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 45 and in Figure 46.

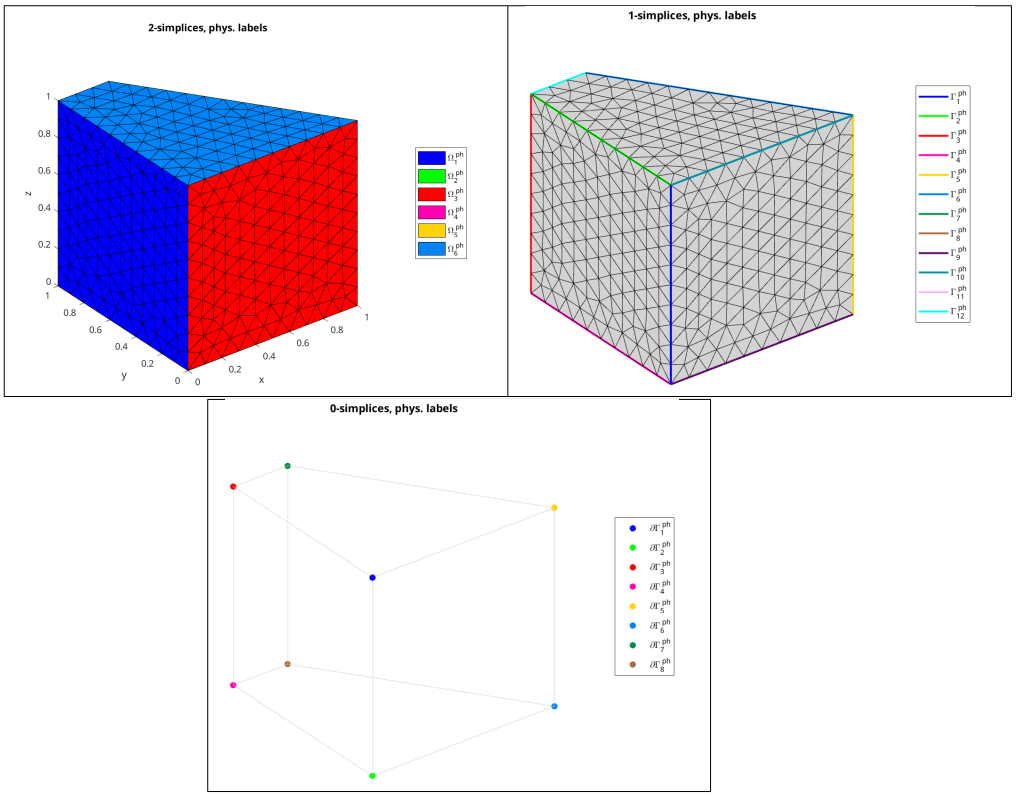


Figure 45: siMesh object corresponding to the surface of the wedge with parameters d = 2, X = 0, Y = 0, Z = 0, DX = 1, DY = 1, DZ = 1, topDX = 0.3. Representation of the physical labels.

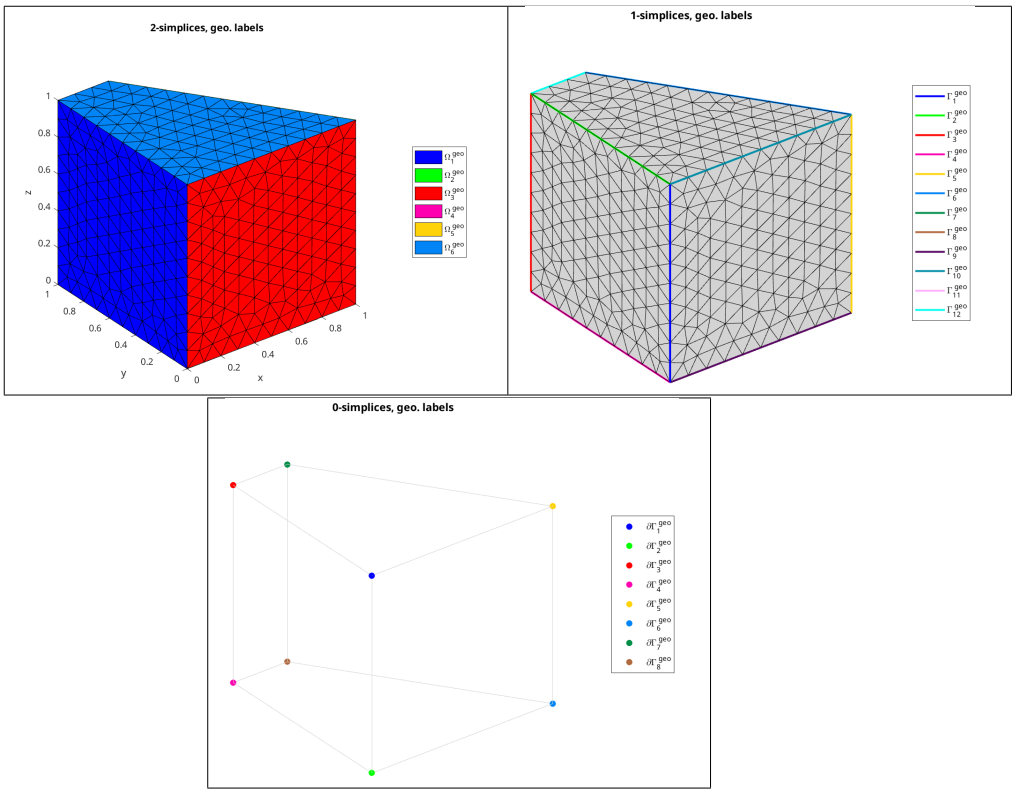


Figure 46: siMesh object corresponding to the surface of the wedge with parameters d = 2, X = 0, Y = 0, Z = 0, DX = 1, DY = 1, DZ = 1, topDX = 0.3. Representation of the geometrical labels.

4.2.8 fc_simesh.samples.torus function

The **fc_simesh**.samples.torus function returns an siMesh object corresponding to a torus obtain by using `gmsh` with `torusOC.geo` file. The torus is defined by

- the 3 coordinates of the center of the first circular face (X, (Y, (Z), default $(0, 0, 0)$,
- two radii R1, default 0.5, R2, default 0.2,
- the angular opening theta $\in]0, 2 * \pi]$, default 2π.

Syntaxe

```
Th=fc_simesh.samples.torus(N)
Th=fc_simesh.samples.torus(N,Name,Value, ...)
```

Description

fc_simesh.samples.torus(N) returns the default torus as a siMesh object where N is a refinement parameter.

fc_simesh.samples.torus(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, d $\in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'X' : to specify X value (default 0),
- 'Y' : to specify Y value (default 0),
- 'Z' : to specify Z value (default 0),
- 'R1' : to specify R1 value (default 0.5)
- 'R2' : to specify R2 value (default 0.2)
- 'theta' : to specify theta value (default 2π)
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.torus(40);
```

The physical and geometrical labels of the **fc_simesh**.siMesh object Th are respectively represented in Figure 47 and in Figure 48.

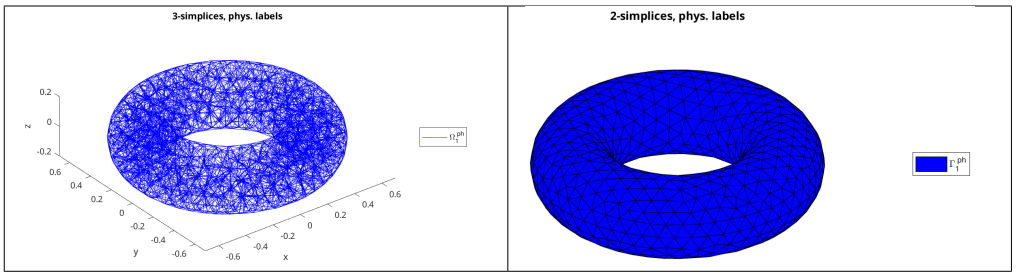


Figure 47: siMesh object corresponding to the default torus. Representation of the physical labels.

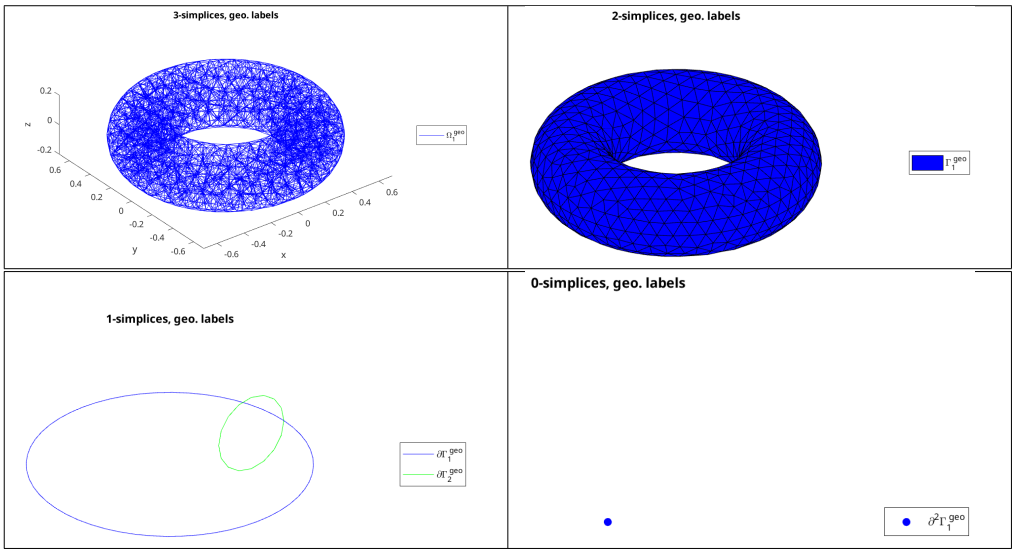


Figure 48: siMesh object corresponding to the default torus. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.torus(40,'X',-1,'Y',2,'Z',0,'R1',1,'R2',0.3,'theta',3*pi/2);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 49 and in Figure 50.

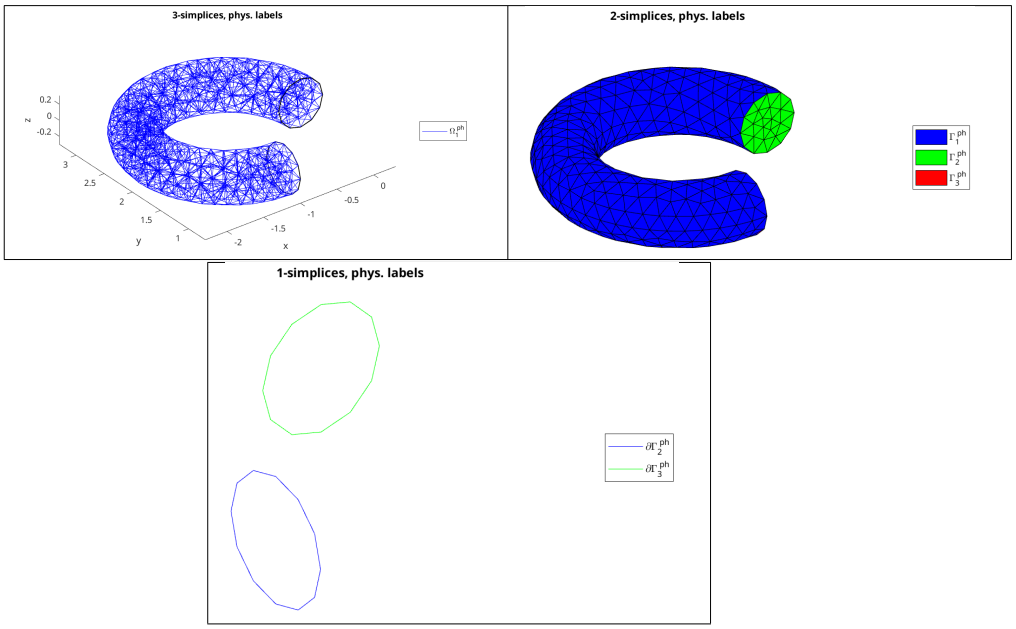


Figure 49: siMesh object corresponding to the torus with parameters X $= -1$, Y $= 2$, Z $= 0$, DX $= 0$, DY $= 3$, DZ $= 0$, R $= 1$ and theta $= \frac{3\pi}{2}$. Representation of the physical labels.

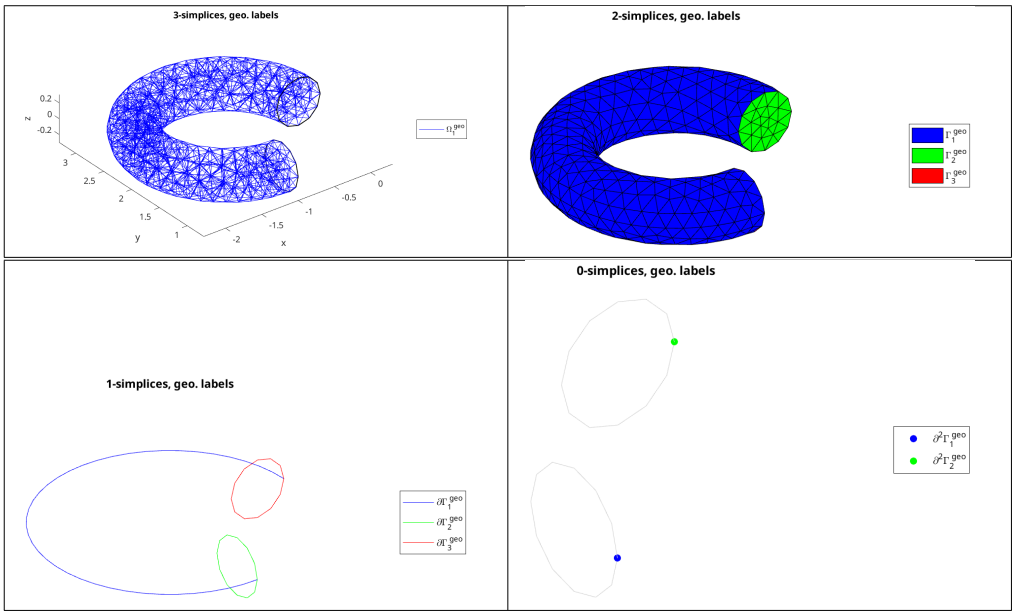


Figure 50: siMesh object corresponding to the torus with parameters X = −1, Y = 2, Z = 0, DX = 0, DY = 3, DZ = 0, R = 1 and theta = $\frac{3\pi}{2}$. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.torus(50,'d',2,'R1',1,'R2',0.2,'theta',3*pi/4);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 51 and in Figure 52.

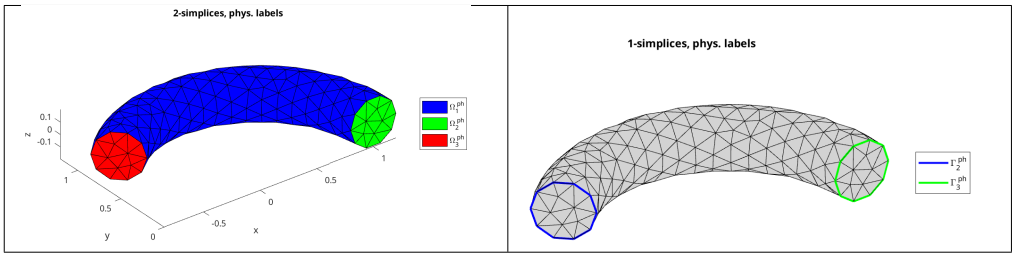


Figure 51: siMesh object corresponding to the surface of the torus with parameters d = 2, X = 0, Y = 0, Z = 0, DX = 1, DY = 0.5, DZ = 1, R1 = 0.5, R2 = 0, and theta = $\frac{3\pi}{4}$.. Representation of the physical labels.

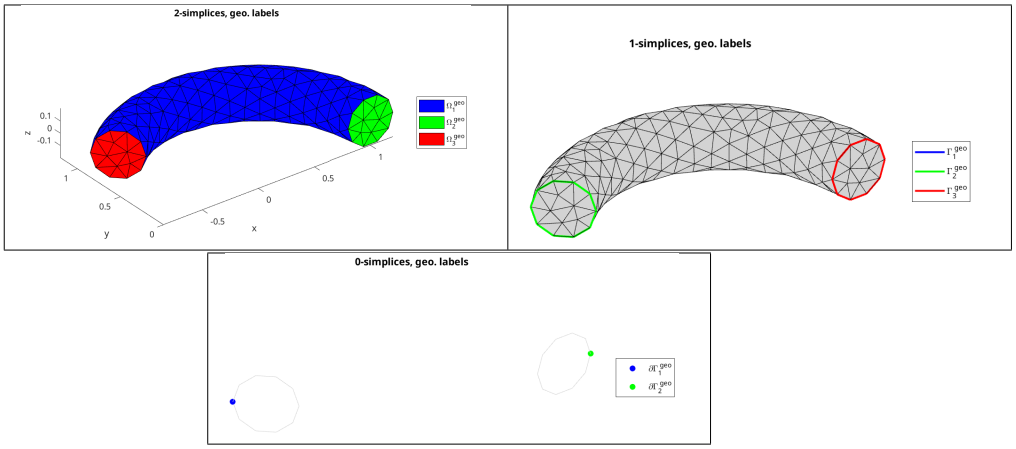


Figure 52: siMesh object corresponding to the surface of the torus with parameters d = 2, X = 0, Y = 0, Z = 0, DX = 1, DY = 0.5, DZ = 1, R1 = 0.5, R2 = 0, and theta = $\frac{3\pi}{4}$.. Representation of the geometrical labels.

4.2.9 fc_simesh.samples.ladder function

The **fc_simesh**.samples.ladder function returns an siMesh object corresponding to a ladder obtain by using `gmsh` with `ladderOC.geo` file. The ladder is made of NB cylindrical rungs of radius Rr and length W. The two slide rail are cylindrical wity radius Rs and with length L. The default parameters are L = 500, Rs = 2, W = 50, Rr = 1, NB = 10.

Syntaxe

```
Th=fc_simesh.samples.ladder(N)
Th=fc_simesh.samples.ladder(N,Name,Value, ...)
```

Description

fc_simesh.samples.ladder(N) returns the ladderwith default parameters as a siMesh object where N is a refinement parameter.

fc_simesh.samples.ladder(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, $d \in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'L' : to specify L value (default 500),
- 'Rs' : to specify Rs value (default 2),
- 'W' : to specify W value (default 40),
- 'Rr' : to specify Rr value (default 1),
- 'NB' : to specify NB value (default 10),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.ladder(5)
```

The physical and geometrical labels of the **fc_simesh**.siMesh object Th are respectively represented in Figure 53 and in Figure 54.

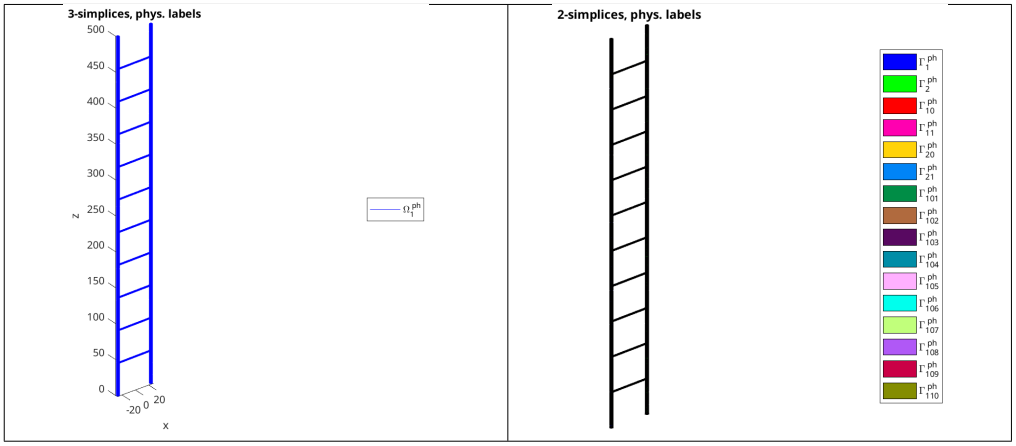


Figure 53: siMesh object corresponding to the default ladder. Representation of the physical labels.

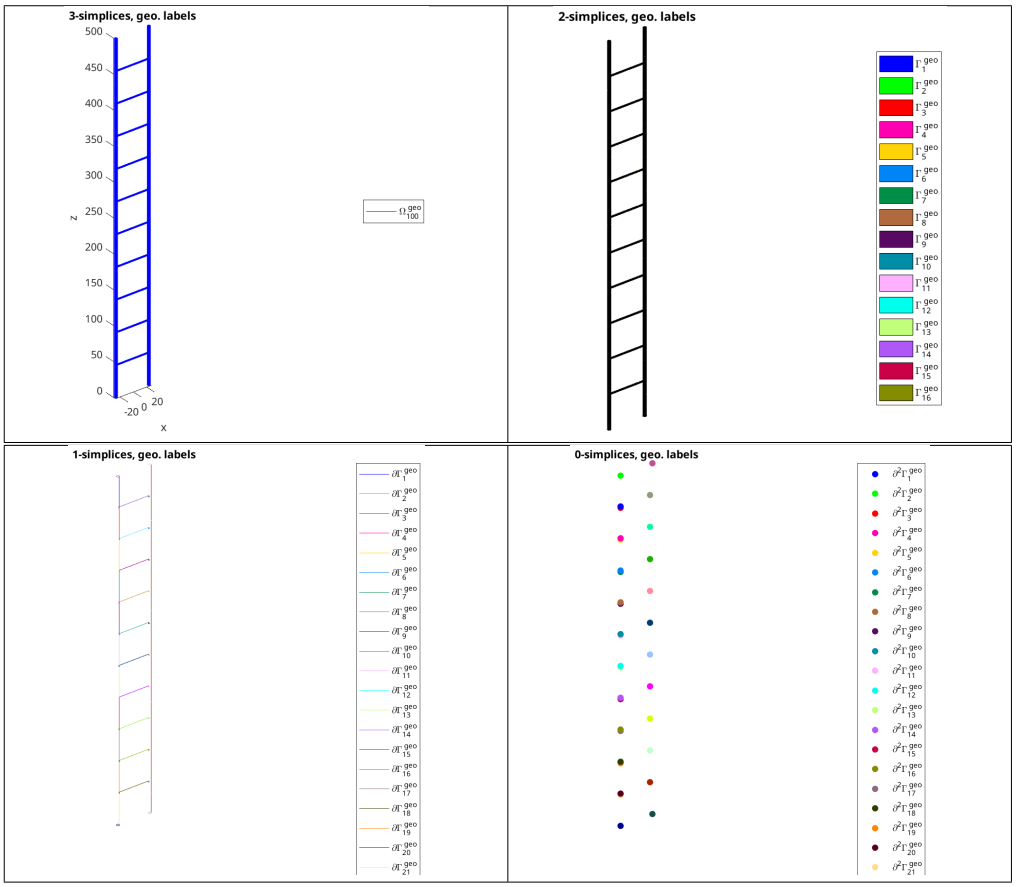


Figure 54: siMesh object corresponding to the default ladder. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.ladder(5,'NB',2,'L',100);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 55 and in Figure 56.

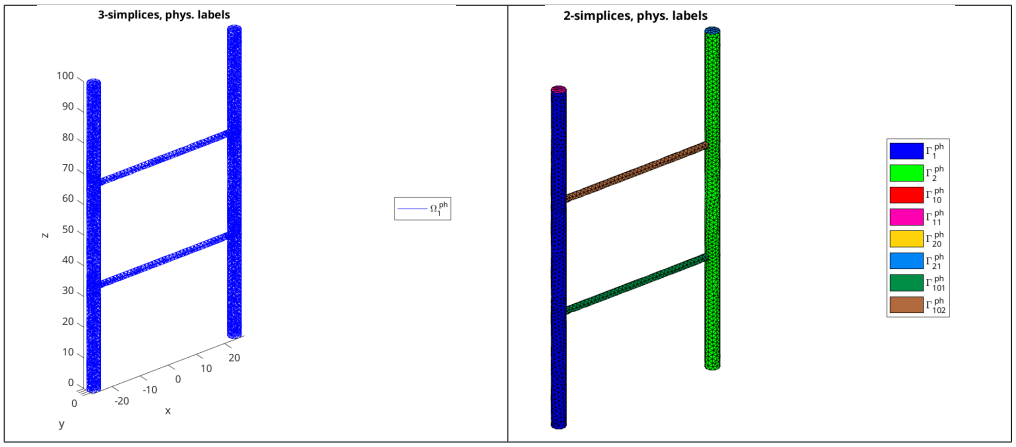


Figure 55: siMesh object corresponding to the ladder with NB = 2 and l = 100. Representation of the physical labels.

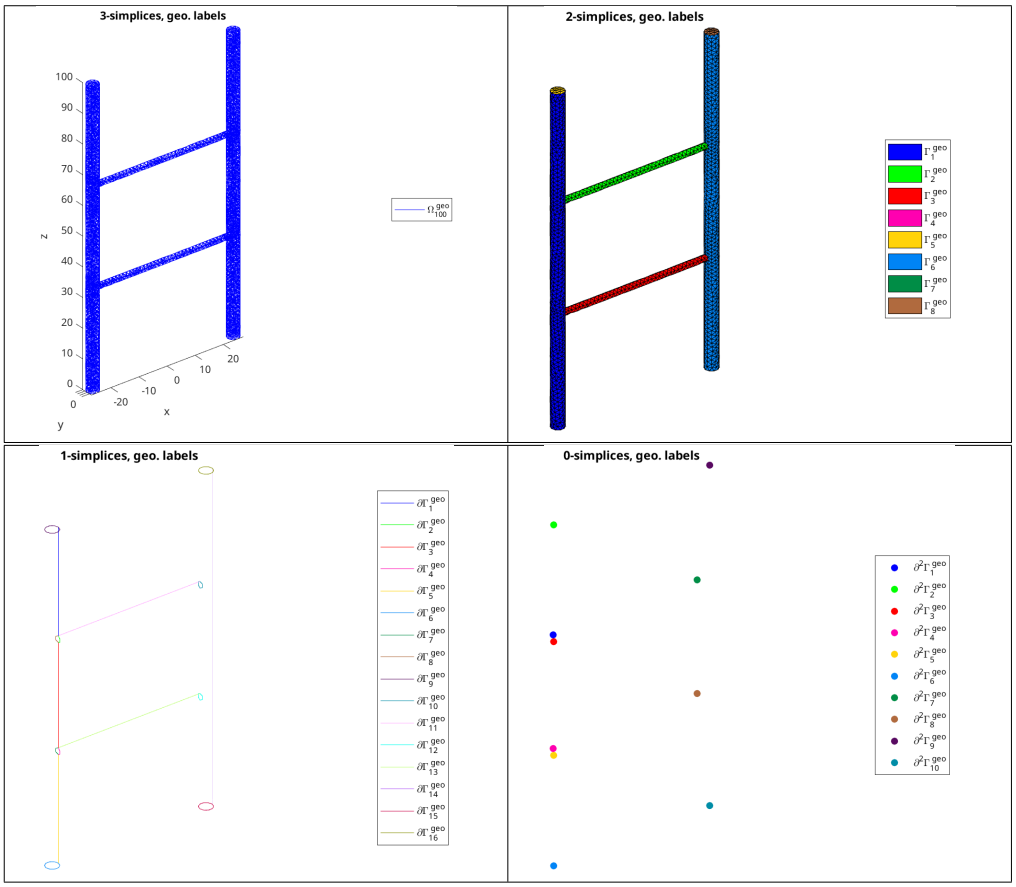


Figure 56: siMesh object corresponding to the ladder with NB = 2 and l = 100. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.ladder(10,'d',2,'NB',2,'L',120,'W',40,'Rs',5,'Rr',5);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 57 and in Figure 58.

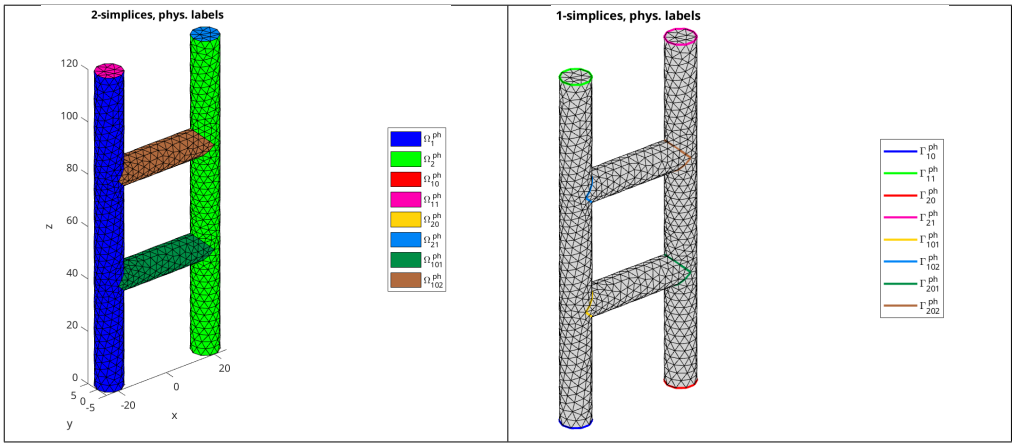


Figure 57: siMesh object corresponding to the surface of the ladder with parameters d = 2, LL = 120, l = 180, RR = 50 and r = 15. Representation of the physical labels.

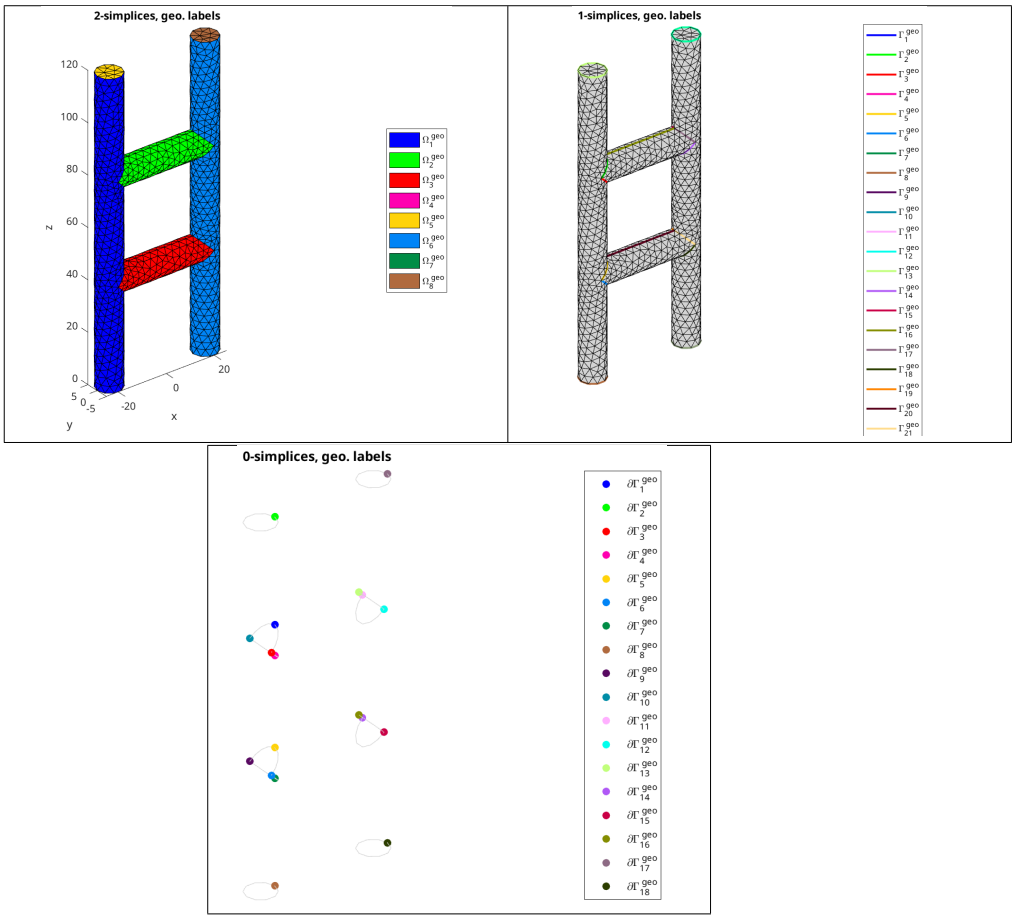


Figure 58: siMesh object corresponding to the surface of the ladder with parameters d = 2, LL = 120, l = 180, RR = 50 and r = 15. Representation of the geometrical labels.

4.2.10 fc_simesh.samples.ladder02 function

The **fc_simesh**.samples.ladder02 function returns an siMesh object corresponding to a ladder obtain by using `gmsh` with `ladder02OC.geo` file. The ladder is made of NB cylindrical rungs of radius Rr and length W. The two slide rail are cylindrical wity radius Rs and with length L. The default parameters are L = 500, Rs = 2, W = 50, Rr = 1, NB = 10.

One cannot get a surface mesh from this function.

Syntaxe

```
Th=fc_simesh.samples.ladder02(N)
Th=fc_simesh.samples.ladder02(N,Name,Value, ...)
```

Description

fc_simesh.samples.ladder02(N) returns the ladderwith default parameters as a siMesh object where N is a refinement parameter.

fc_simesh.samples.ladder02(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'L' : to specify L value (default 500),
- 'Rs' : to specify Rs value (default 2),
- 'W' : to specify W value (default 40),
- 'Rr' : to specify Rr value (default 1),
- 'NB' : to specify NB value (default 10),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.ladder02(5);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 59 and in Figure 60.

Figure 59: siMesh object corresponding to the default ladder. Representation of the physical labels.

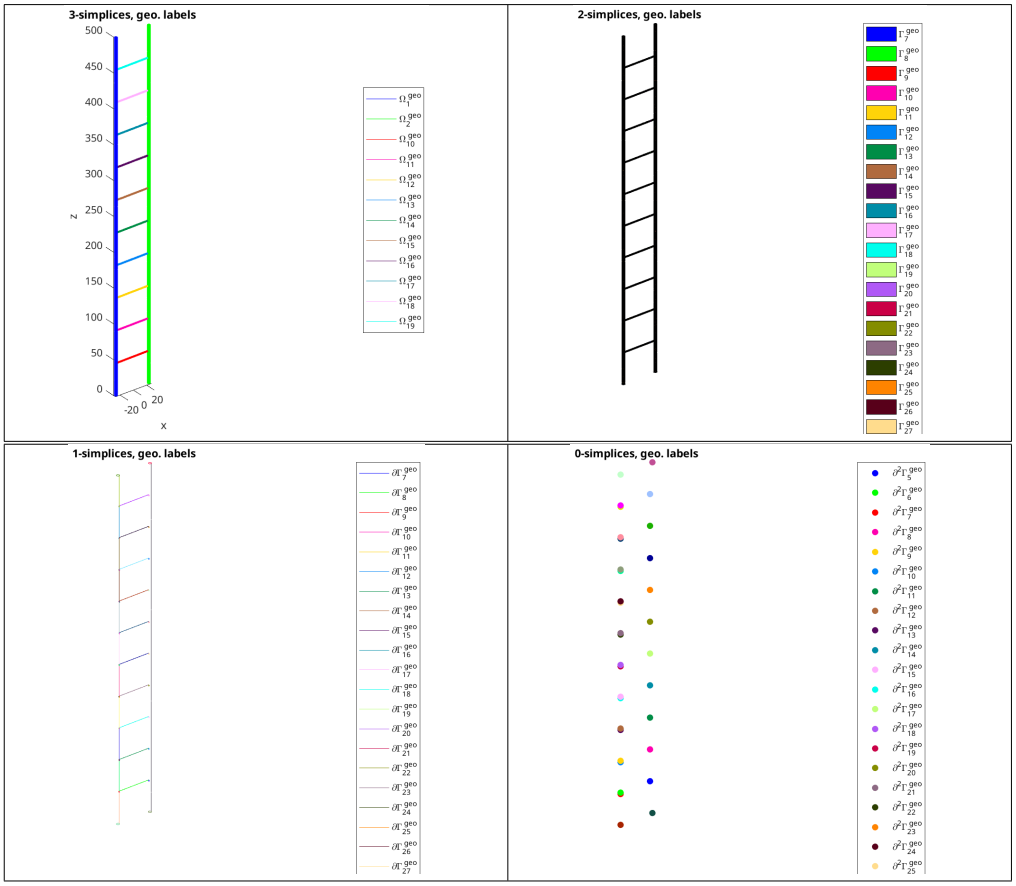


Figure 60: siMesh object corresponding to the default ladder. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.ladder02(9,'NB',2,'L',120,'W',40,'Rs',5,'Rr',5);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 61 and in Figure 62.

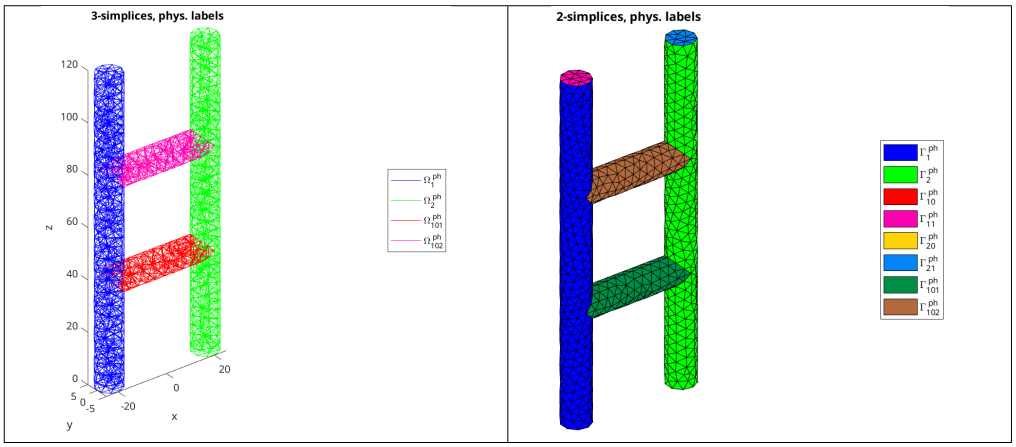


Figure 61: siMesh object corresponding to the ladder with NB = 2, L = 120, W = 40, Rs = 5 and Rr = 5. Representation of the physical labels.

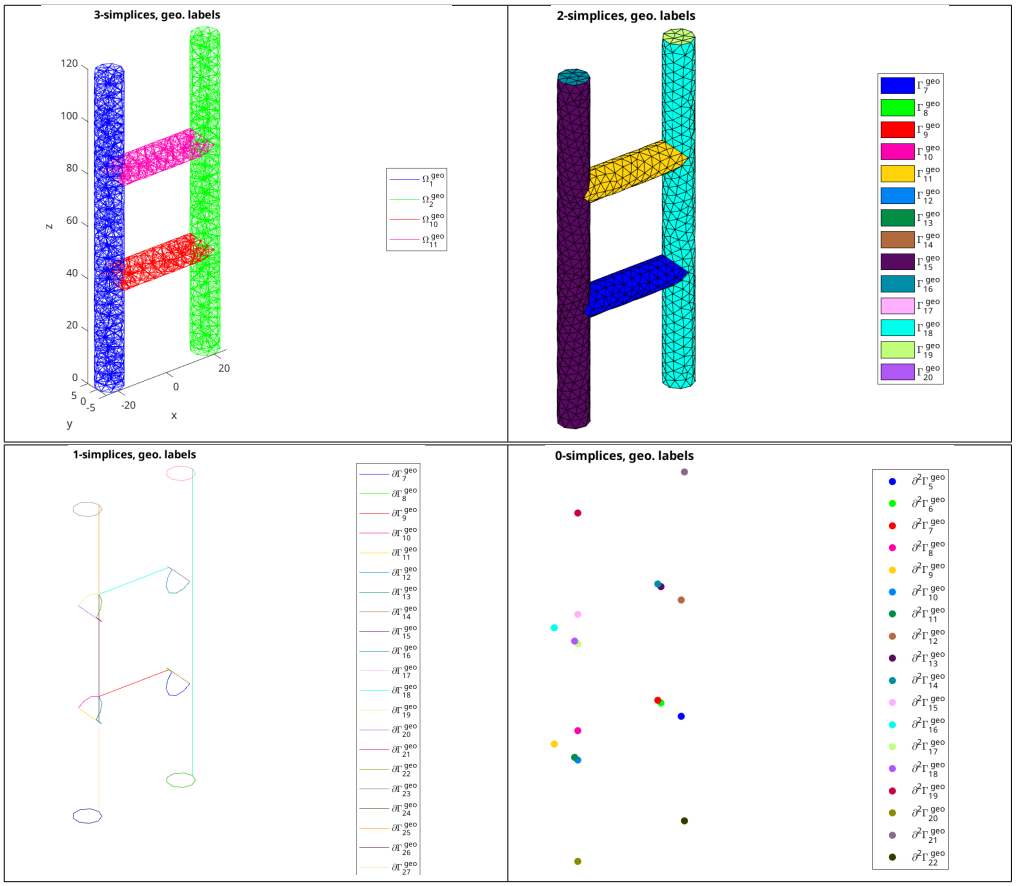


Figure 62: siMesh object corresponding to the ladder with NB = 2, L = 120, W = 40, Rs = 5 and Rr = 5. Representation of the geometrical labels.

4.2.11 fc_simesh.samples.tuning_fork function

The **fc_simesh**.samples.tuning_fork function returns an siMesh object corresponding to a tuning fork obtain by using `gmsh` with `tuning_forkOC.geo` file.

Syntaxe

```
Th=fc_simesh.samples.tuning_fork(N)
Th=fc_simesh.samples.tuning_fork(N,Name,Value, ...)
```

Description

fc_simesh.samples.tuning_fork(N) returns a tuning fork as a siMesh object where N is a refinement parameter. By default, the tuning fork is : to do!

fc_simesh.samples.tuning_fork(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, $d \in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'LL' : to specify LL value (default 120),
- 'l' : to specify l value (default 80),
- 'RR' : to specify LL value (default 40),
- 'r' : to specify l value (default 10),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.tuning_fork(15)
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 63 and in Figure 64.

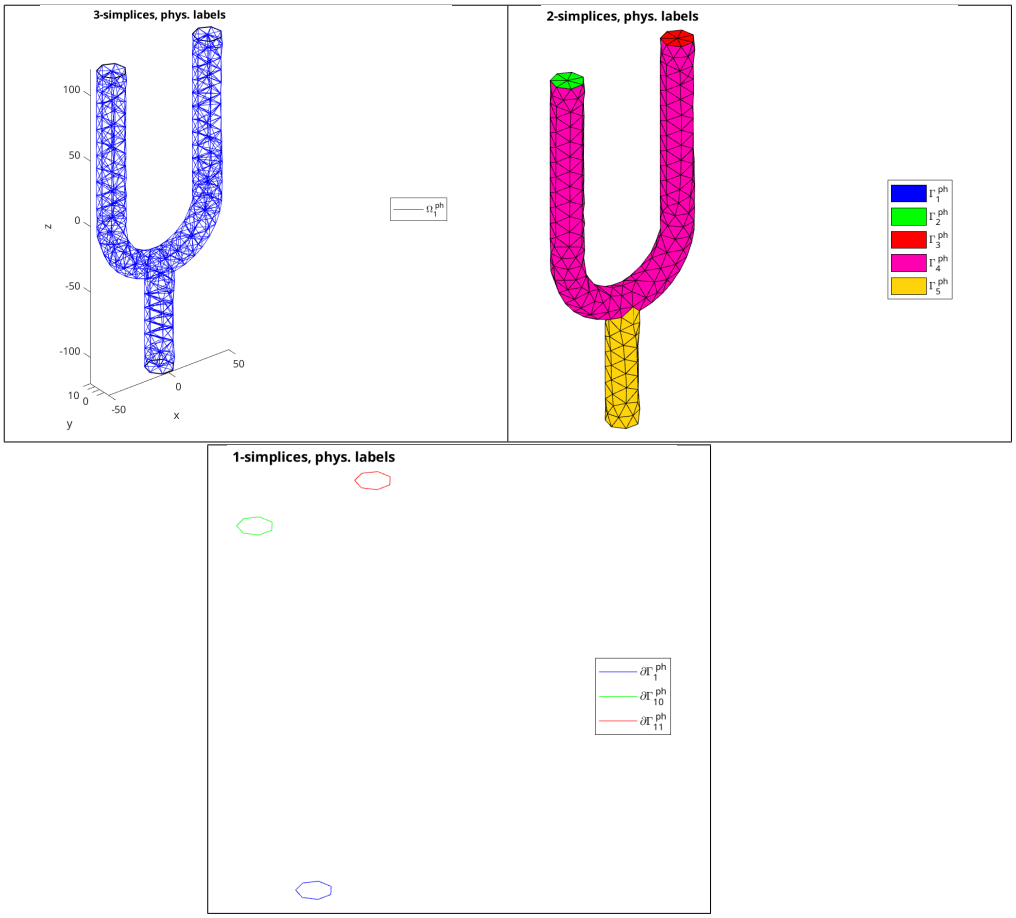


Figure 63: siMesh object corresponding to the default tuning fork. Representation of the physical labels.

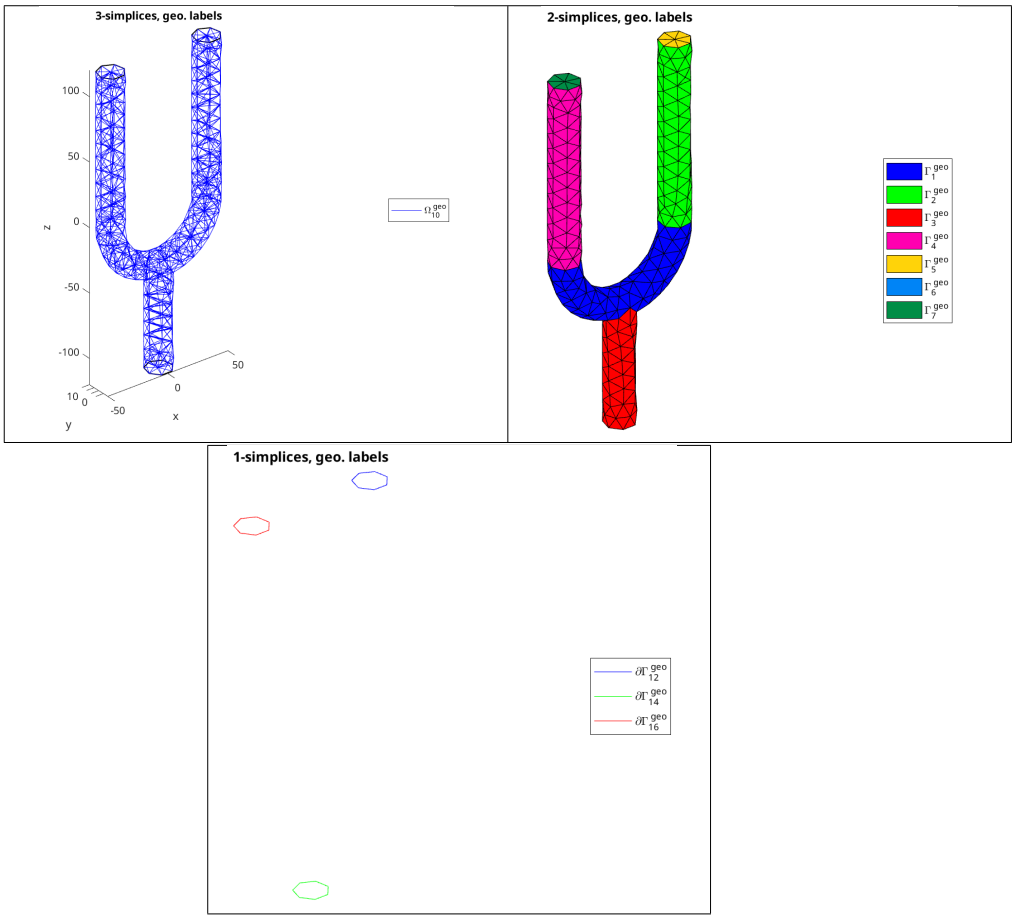


Figure 64: siMesh object corresponding to the default tuning fork. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.tuning_fork(15,'LL',200,'l',50);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 65 and in Figure 66.

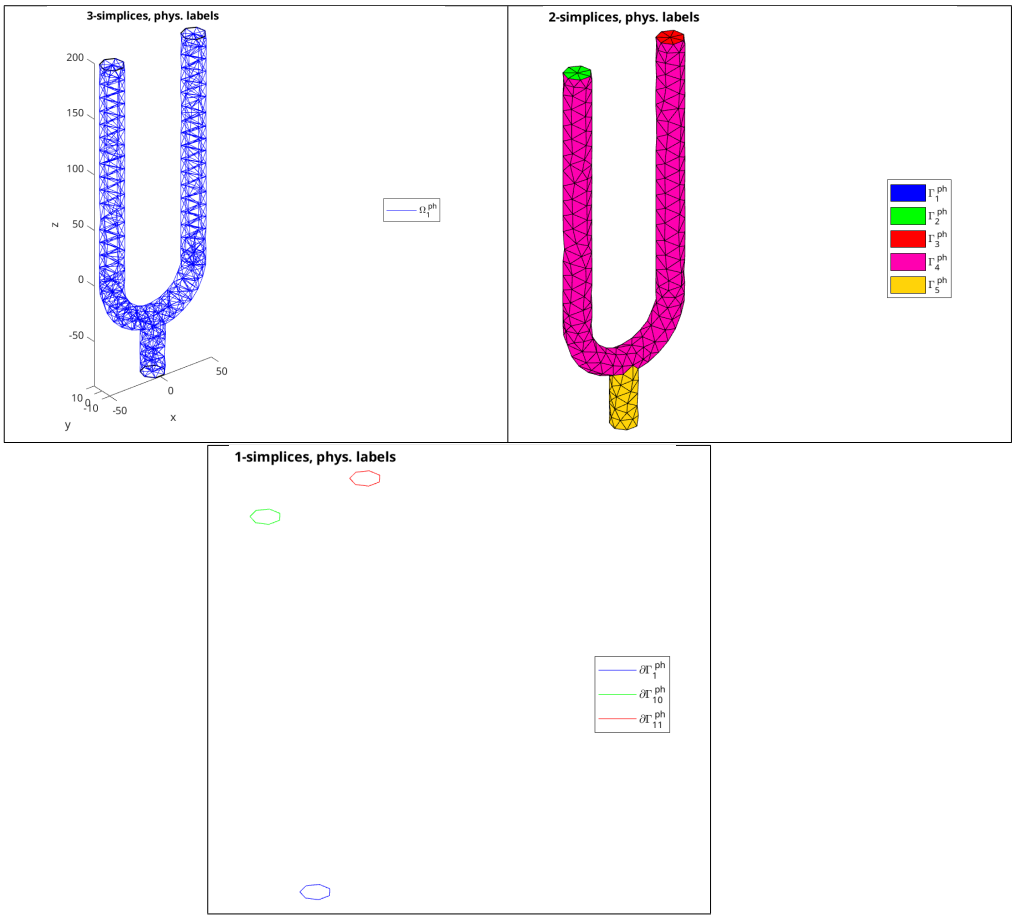


Figure 65: siMesh object corresponding to the tuning fork with LL = 200 and l = 50. Representation of the physical labels.

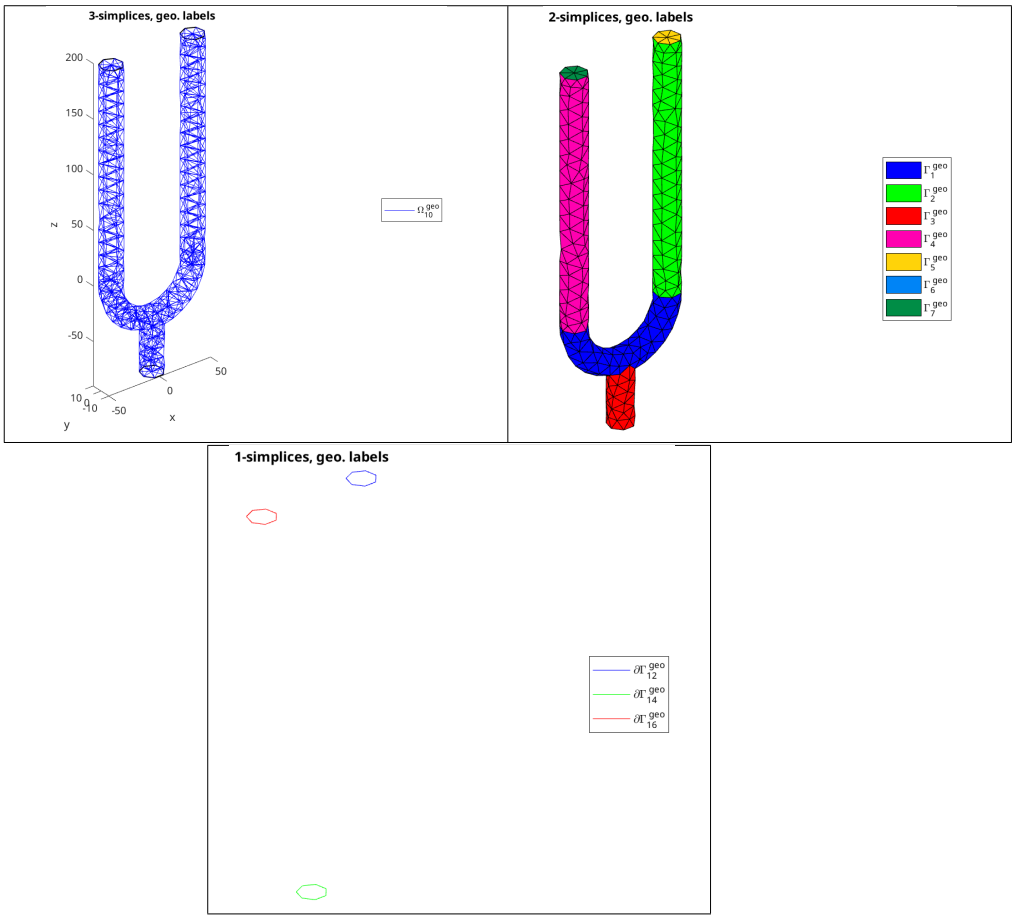


Figure 66: siMesh object corresponding to the tuning fork with LL = 200 and l = 50. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.tuning_fork(15,'d',2,'LL',120,'l',180,'RR',50,'r',15);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 67 and in Figure 68.

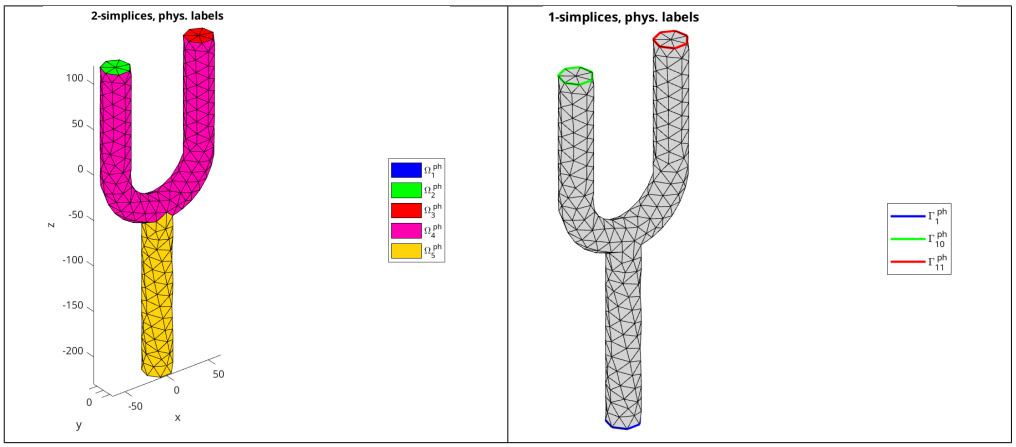


Figure 67: siMesh object corresponding to the surface of the tuning fork with parameters d = 2, LL = 120, l = 180, RR = 50 and r = 15. Representation of the physical labels.

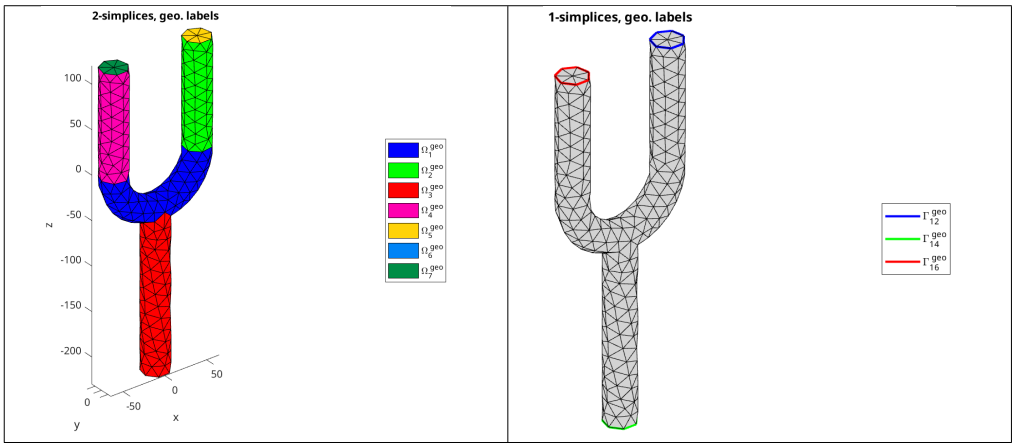


Figure 68: siMesh object corresponding to the surface of the tuning fork with parameters d = 2, LL = 120, l = 180, RR = 50 and r = 15. Representation of the geometrical labels.

4.2.12 fc_simesh.samples.construction01 function

The **fc_simesh**.samples.construction01 function returns an siMesh object obtain by using `gmsh` with `construction01OC.geo` file. The object is made of seven balls of radius R2, one centered in the origin, the others centered on the sphere of radius R1 connected with some torus of small radius r.

Restricted to : r<=R2/2 and R1>=2*R2.

Syntaxe

```
Th=fc_simesh.samples.construction01(N)
Th=fc_simesh.samples.construction01(N,Name,Value, ...)
```

Description

fc_simesh.samples.construction01(N) returns the construction01with default parameters as a siMesh object where N is a refinement parameter.

fc_simesh.samples.construction01(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, $d \in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'R1' : to specify R1 value (default 15),
- 'R2' : to specify R2 value (default 5),
- 'r' : to specify r value (default 1),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.construction01(8)
```

The physical and geometrical labels of the **fc_simesh**.siMesh object Th are respectively represented in Figure 69 and in Figure 70.

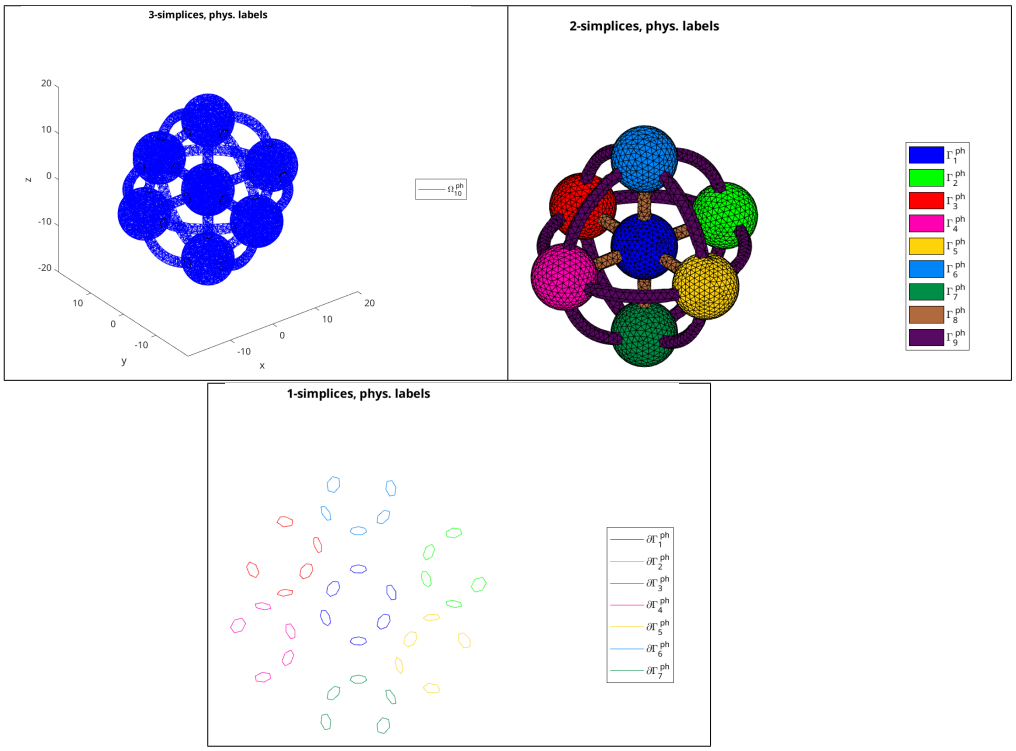


Figure 69: siMesh object corresponding to the default construction01. Representation of the physical labels.

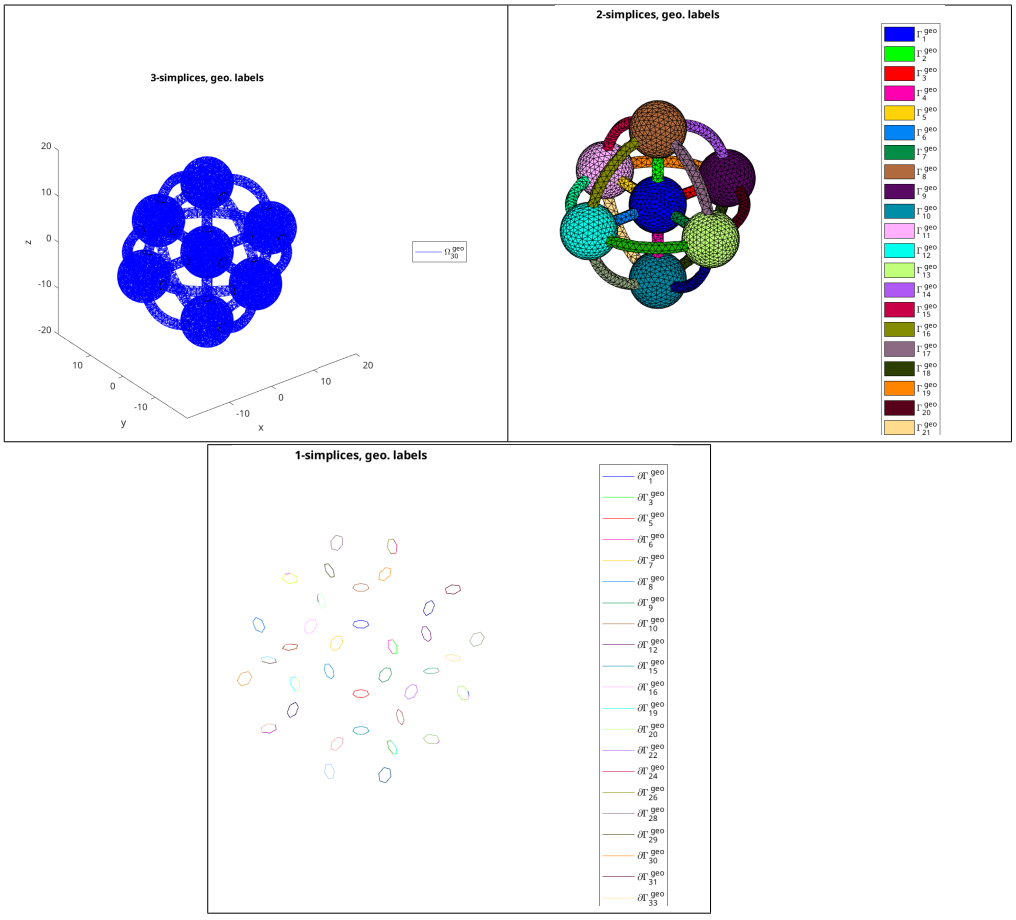


Figure 70: siMesh object corresponding to the default construction01. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.construction01(8,'R1',25,'R2',4,'r',2);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 71 and in Figure 72.

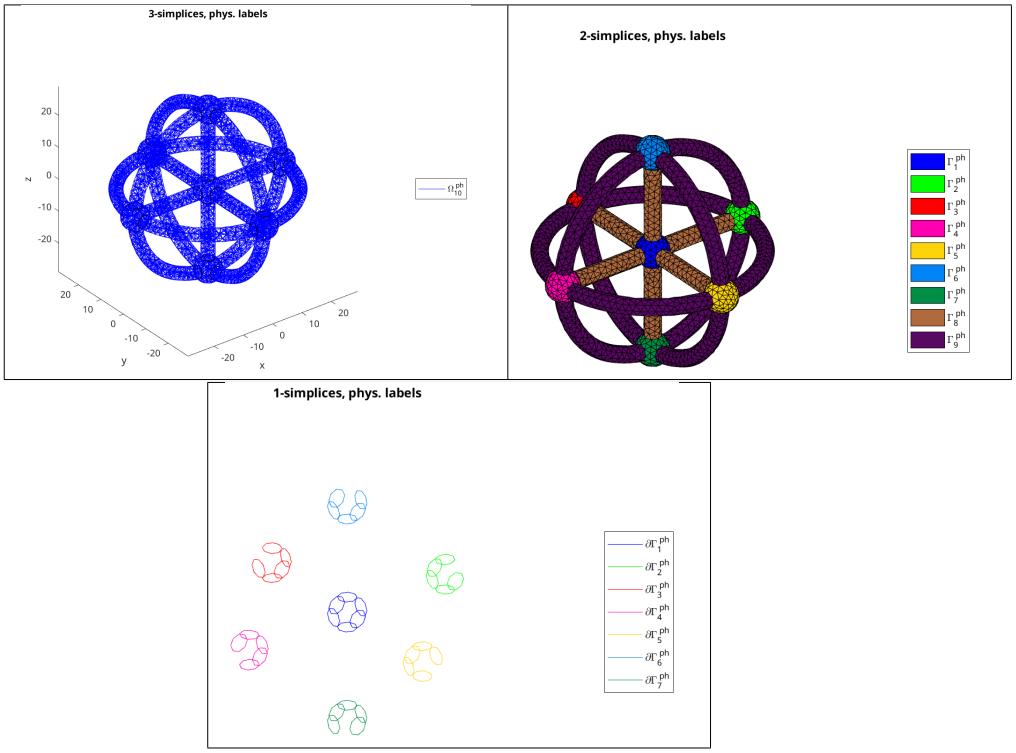


Figure 71: siMesh object corresponding to the construction01 with R1 = 25, R2 = 4 and r = 2. Representation of the physical labels.

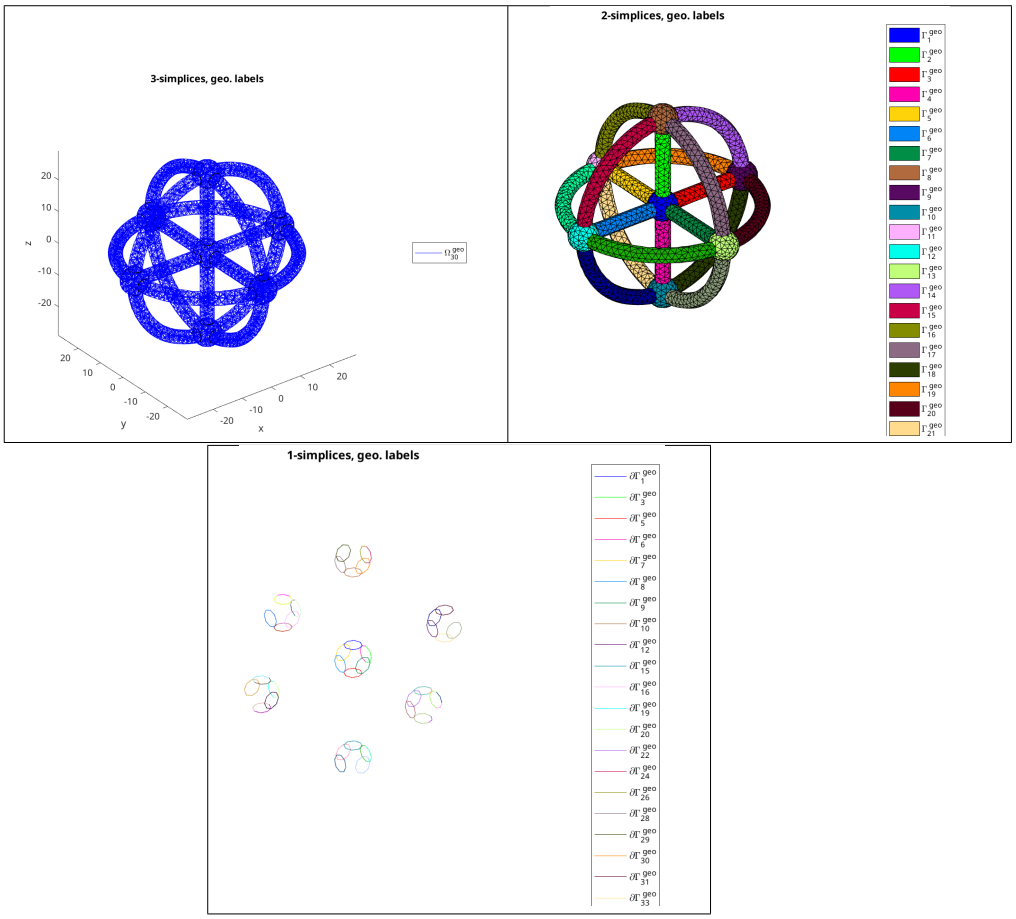


Figure 72: siMesh object corresponding to the construction01 with R1 = 25, R2 = 4 and r = 2. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.construction01(8,'d',2,'R1',25,'R2',8,'r',2);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 73 and in Figure 74.

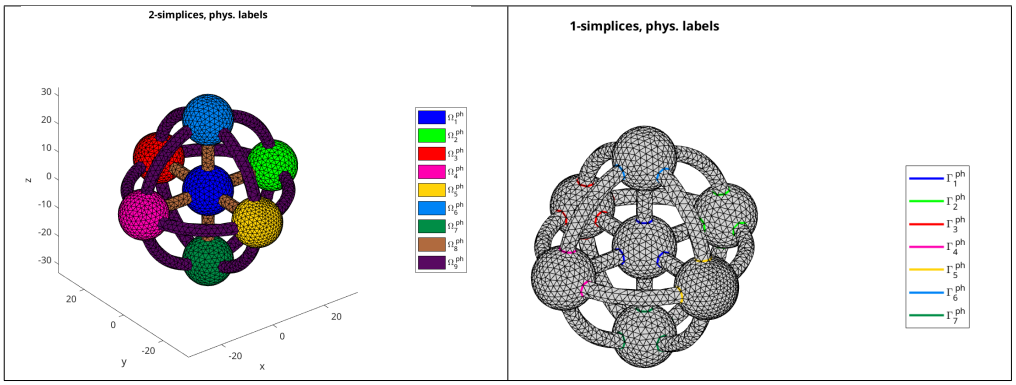


Figure 73: siMesh object corresponding to the surface of the construction01 with parameters d = 2, R1 = 25, R2 = 8 and r = 2. Representation of the physical labels.

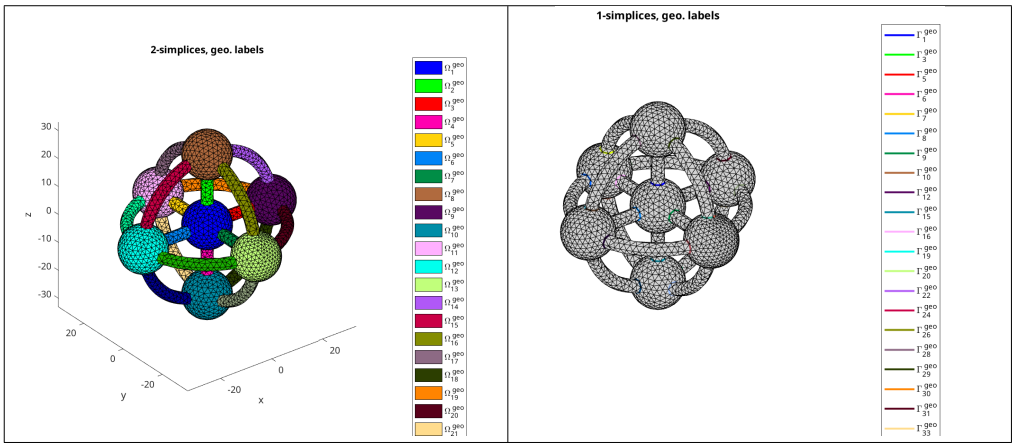


Figure 74: siMesh object corresponding to the surface of the construction01 with parameters d = 2, R1 = 25, R2 = 8 and r = 2. Representation of the geometrical labels.

4.2.13 fc_simesh.samples.construction02 function

The **fc_simesh**.samples.construction02 function returns an siMesh object obtain by using `gmsh` with `construction02OC.geo` file. The object is made of seven balls of radius R2, one centered in the origin, the others centered on the sphere of radius R1 connected with some torus of small radius r. The object can be dilated in each direction by using dX, dY, dZ parameters (all defaults are 1).

Restricted to : r<=R2/2 and R1>=2*R2.

Syntaxe

```
Th=fc_simesh.samples.construction02(N)
Th=fc_simesh.samples.construction02(N,Name,Value, ...)
```

Description

fc_simesh.samples.construction02(N) returns an object with default parameters as a siMesh object where N is a refinement parameter.

fc_simesh.samples.construction02(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, $d \in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'R1' : to specify R1 value (default 15),
- 'R2' : to specify R2 value (default 5),
- 'r' : to specify r value (default 1),
- 'dX' : to specify dX value (default 1),
- 'dY' : to specify dX value (default 1),
- 'dZ' : to specify dX value (default 1),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.construction02(8)
```

The physical and geometrical labels of the **fc_simesh**.siMesh object Th are respectively represented in Figure 75 and in Figure 76.

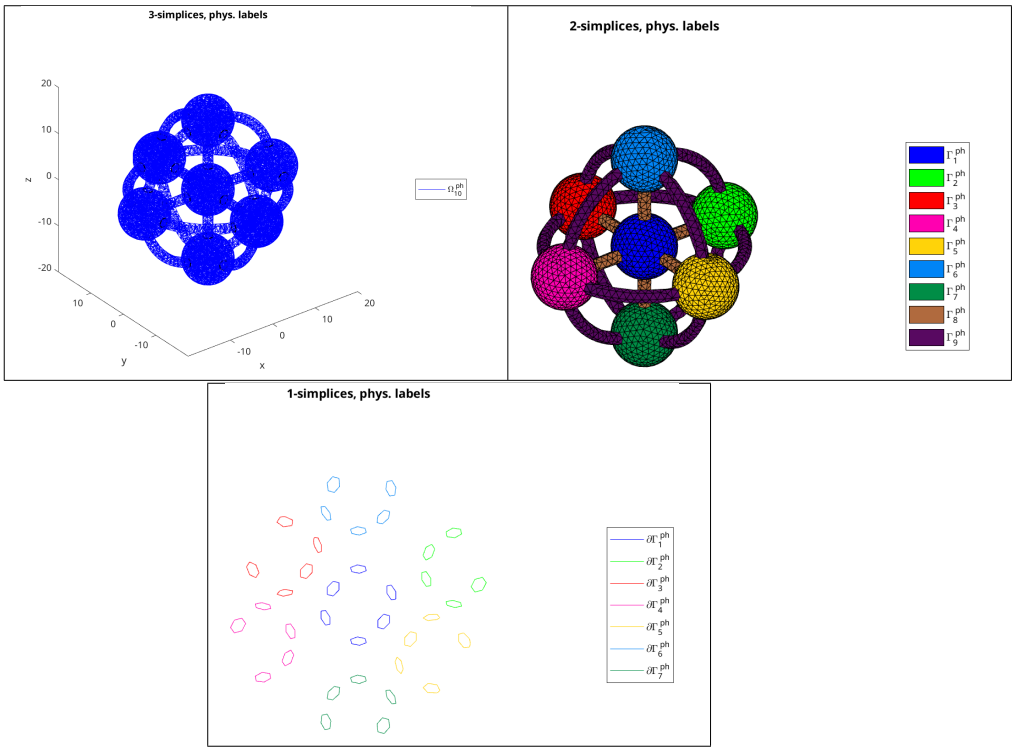


Figure 75: siMesh object corresponding to the default `construction02` object. Representation of the physical labels.

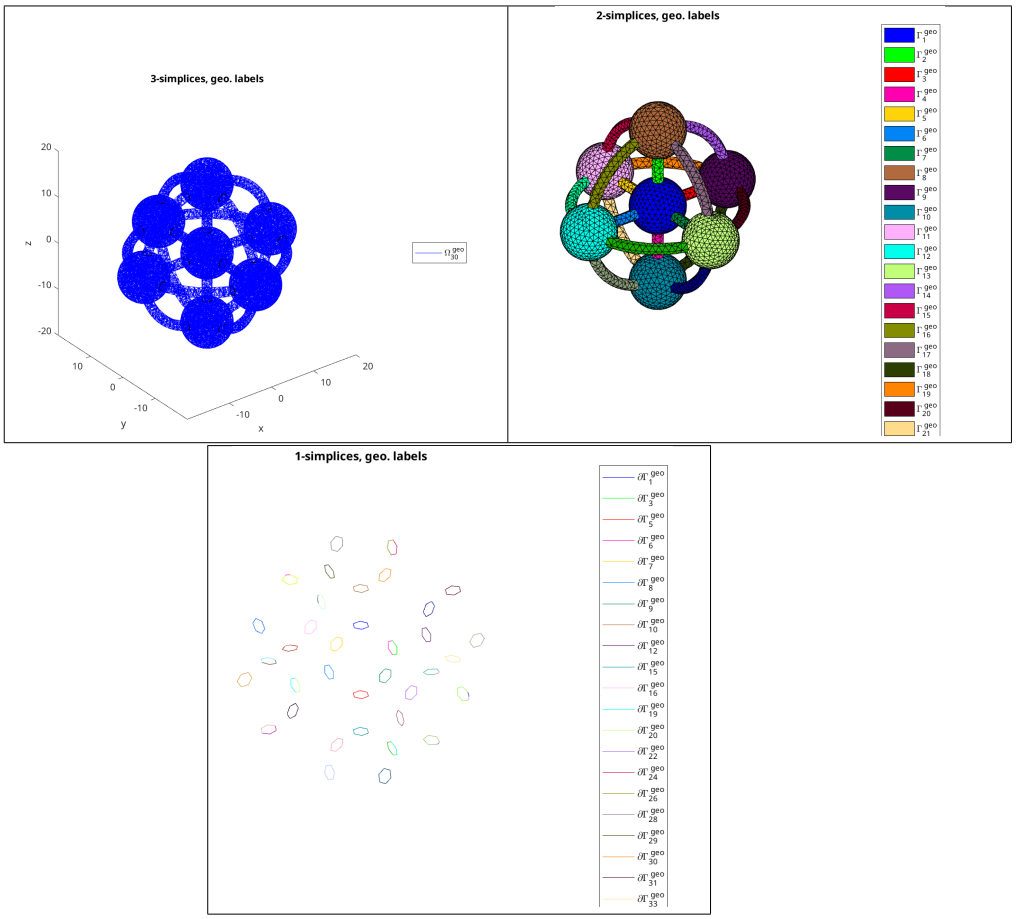


Figure 76: siMesh object corresponding to the default `construction02` object. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.construction02(8,'R1',25,'R2',8,'r',2,'dZ',1.5);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 77 and in Figure 78.

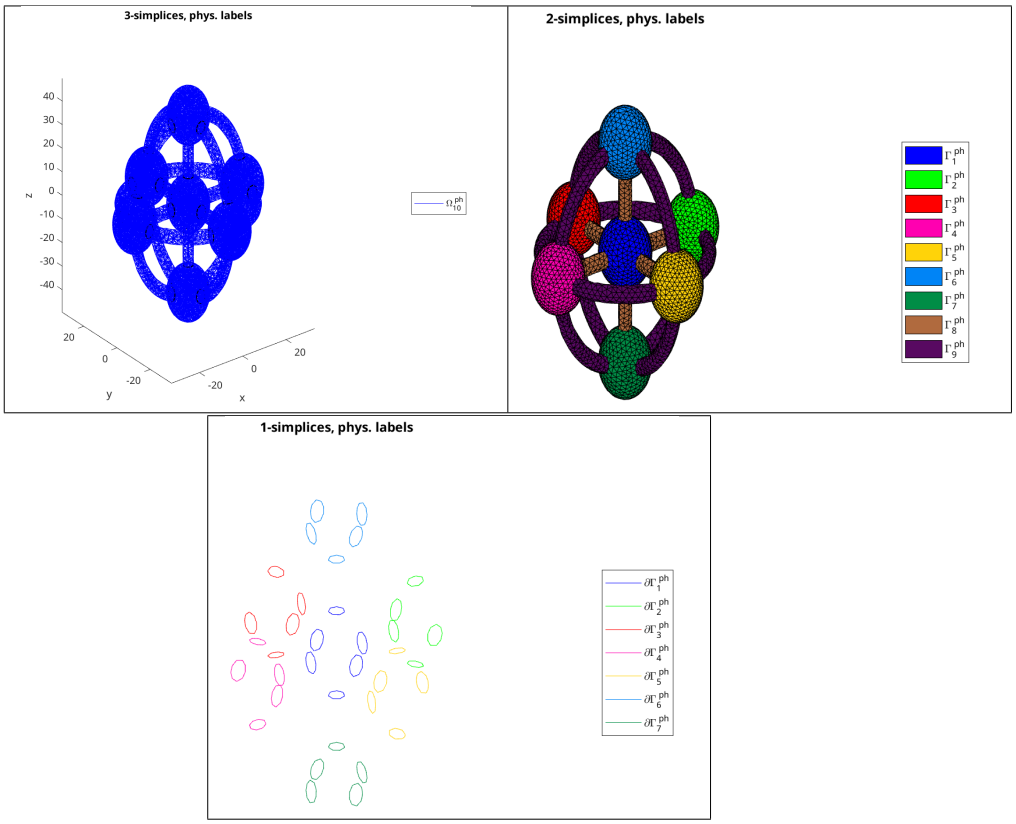


Figure 77: siMesh object corresponding to the `construction02` object with R1 = 25, R2 = 8, r = 2 and dZ = 1.5. Representation of the physical labels.

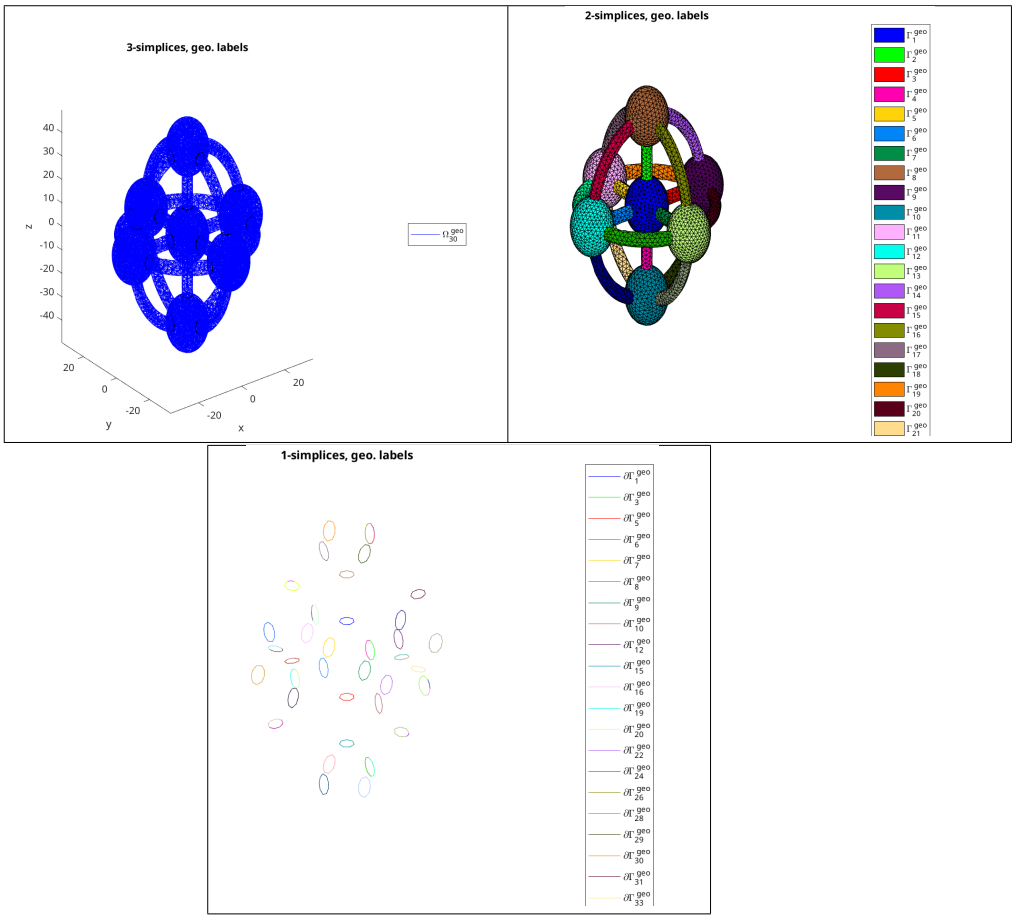


Figure 78: siMesh object corresponding to the `construction02` object with R1 = 25, R2 = 8, r = 2 and dZ = 1.5. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.construction02(8,'d',2,'R1',25,'R2',7,'r',3,'dZ',0.75,'dX',1.5,'dY',1.5);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 79 and in Figure 80.

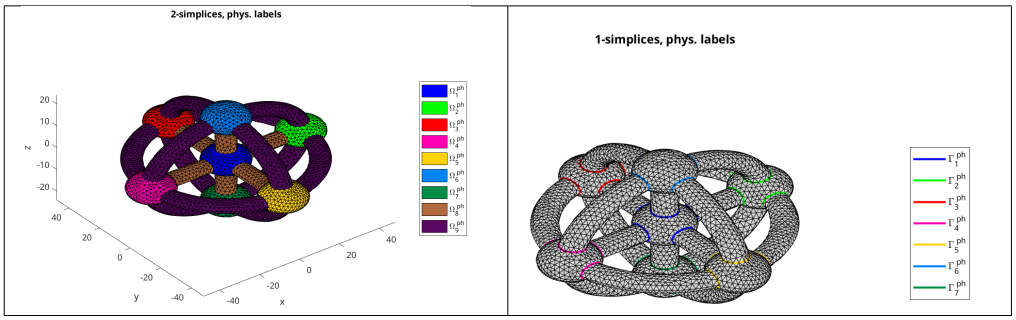


Figure 79: siMesh object corresponding to the surface of the `construction02` object with parameters d = 2, R1 = 25, R2 = 7, r = 3, dX = 1.5, dY = 1.5 and dZ = 0.75. Representation of the physical labels.

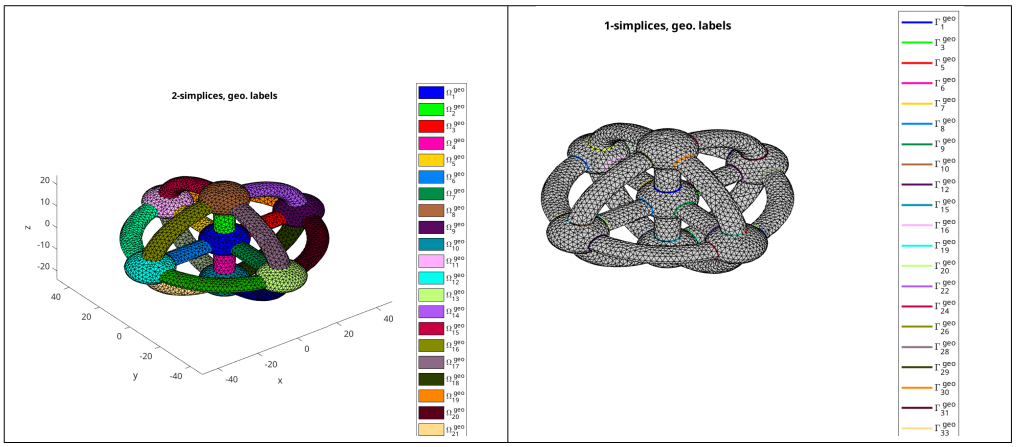


Figure 80: siMesh object corresponding to the surface of the `construction02` object with parameters d = 2, R1 = 25, R2 = 7, r = 3, dX = 1.5, dY = 1.5 and dZ = 0.75. Representation of the geometrical labels.

4.2.14 fc_simesh.samples.spatial function

The **fc_simesh**.samples.spatial function returns an siMesh object obtain by using `gmsh` with `spatialOC.geo` file. The object is made of seven balls of radius R2, one centered in the origin, the others centered on the sphere of radius R1 connected with some torus of small radius r. The object can be dilated in each direction by using dX, dY, dZ parameters (all defaults are 1).

Restricted to : r<=R2/2 and R1>=2*R2.

Syntaxe

```
Th=fc_simesh.samples.spatial(N)
Th=fc_simesh.samples.spatial(N,Name,Value, ...)
```

Description

fc_simesh.samples.spatial(N) returns an object with default parameters as a siMesh object where N is a refinement parameter.

fc_simesh.samples.spatial(N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. Options are

- 'd' : to specify d value, $d \in \{2, 3\}$, volume mesh (3 default) or surface mesh (2),
- 'R1' : to specify R1 value (default 15),
- 'R2' : to specify R2 value (default 5),
- 'r' : to specify r value (default 1),
- 'dX' : to specify dX value (default 1),
- 'dY' : to specify dX value (default 1),
- 'dZ' : to specify dX value (default 1),
- 'verbose' : to specify verbosity from 0 to 4...
- 'delete' : if true, deletes generated mesh file (default : **true**)

Sample 1 (volume mesh)

```
Th=fc_simesh.samples.spatial(8)
```

The physical and geometrical labels of the **fc_simesh**.siMesh object Th are respectively represented in Figure 81 and in Figure 82.

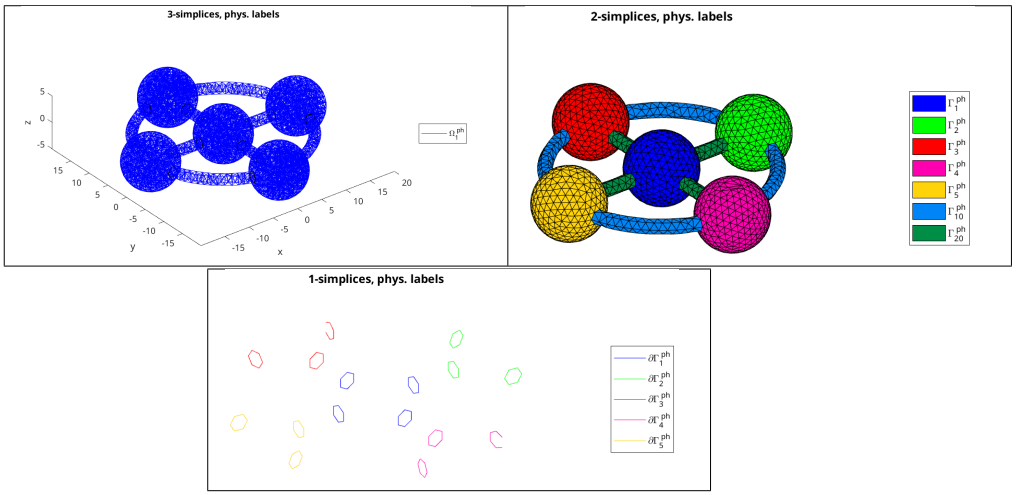


Figure 81: siMesh object corresponding to the default `spatial` object. Representation of the physical labels.

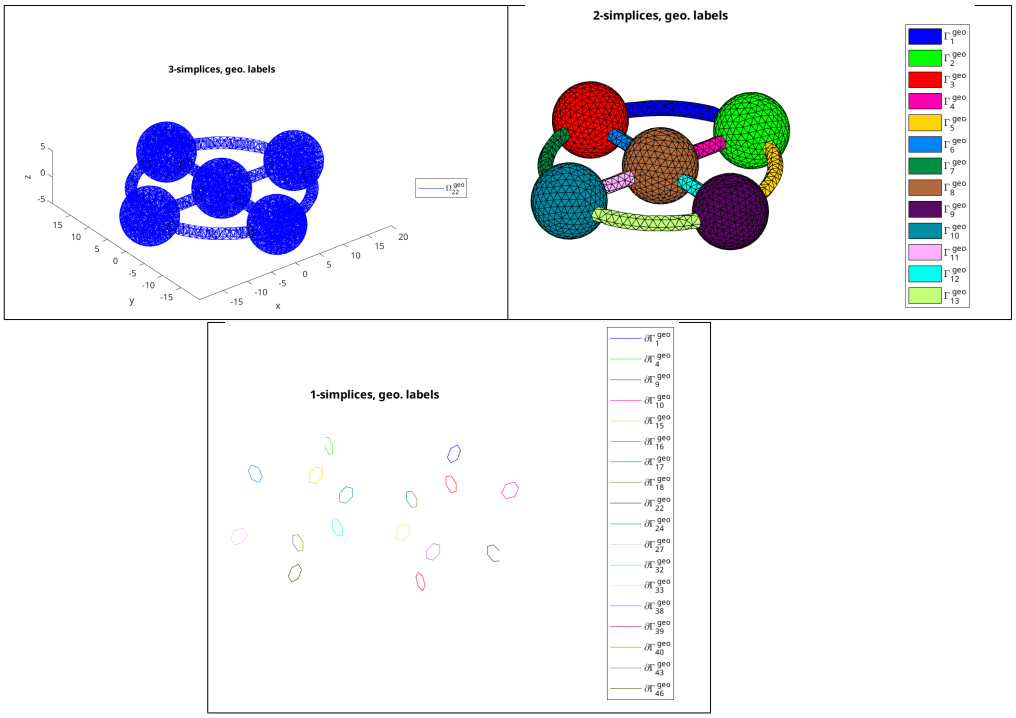


Figure 82: siMesh object corresponding to the default `spatial` object. Representation of the geometrical labels.

Sample 2 (volume mesh)

```
Th=fc_simesh.samples.spatial(8,'R1',25,'R2',8,'r',2,'dY',0.8);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 83 and in Figure 84.

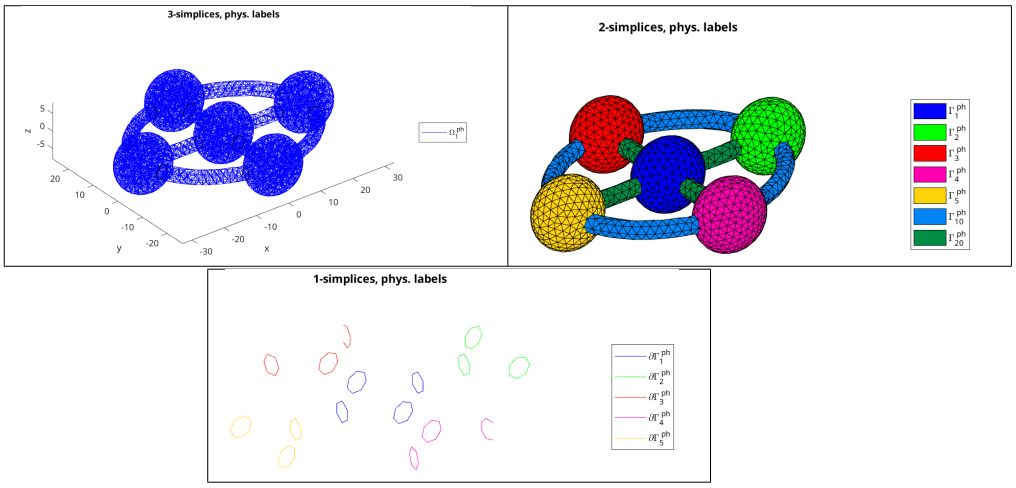


Figure 83: siMesh object corresponding to the `spatial` object with R1 = 25, R2 = 8, r = 2 and dY = 0.8. Representation of the physical labels.

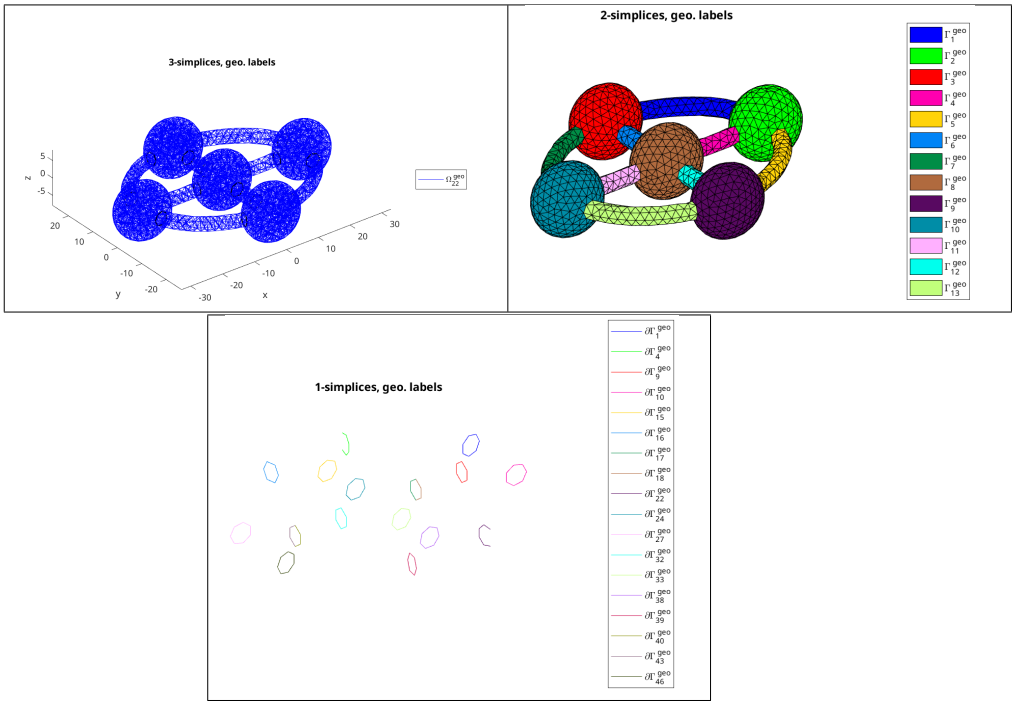


Figure 84: siMesh object corresponding to the `spatial` object with R1 = 25, R2 = 8, r = 2 and dY = 0.8. Representation of the geometrical labels.

Sample 3 (surface mesh)

```
Th=fc_simesh.samples.spatial(8,'d',2,'R1',25,'R2',7,'r',3,'dZ',0.75,'dX',1.5,'dY',0.75);
```

The physical and geometrical labels of the **fc_simesh.siMesh** object Th are respectively represented in Figure 85 and in Figure 86.

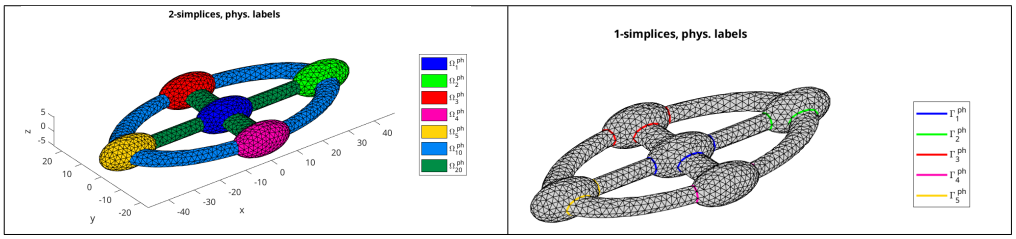


Figure 85: siMesh object corresponding to the surface of the `spatial` object with parameters d = 2, R1 = 25, R2 = 7, r = 3, dX = 1.5, dY = 1.5 and dZ = 0.75. Representation of the physical labels.

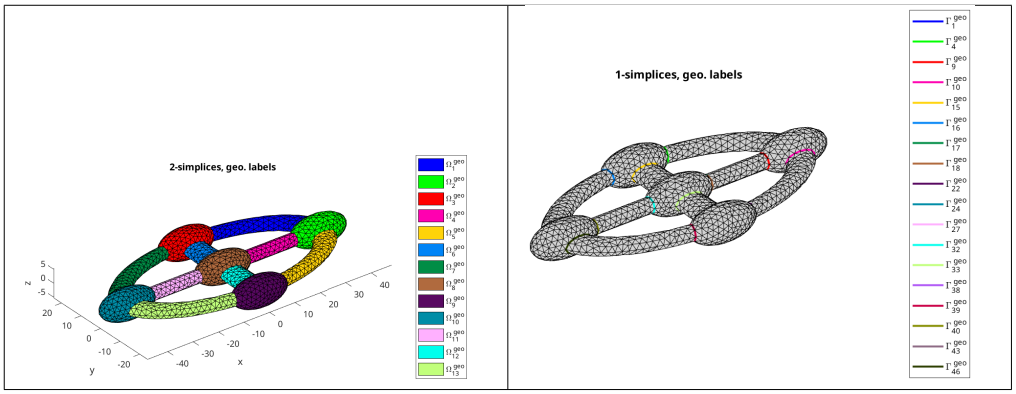


Figure 86: siMesh object corresponding to the surface of the `spatial` object with parameters d = 2, R1 = 25, R2 = 7, r = 3, dX = 1.5, dY = 1.5 and dZ = 0.75. Representation of the geometrical labels.

Appendices

A Listings

B References

[1] F. Cuvelier. fc_oogmsh: an object-oriented Matlab toolbox to run **gmsh** and read mesh files. http://www.math.univ-paris13.fr/~cuvelier/software/, 2017. User's Guide.

[2] F. Cuvelier. fc_siplt: an add-on to the fc_simesh Matlab toolbox for displaying simplices meshes or datas on simplices meshes. http://www.math.univ-paris13.fr/~cuvelier/software/, 2017. User's Guide.

[3] F. Cuvelier. fc_hypermesh: a object-oriented Matlab toolbox to mesh any d-orthotopes (hyperrectangle in dimension d) and their m-faces with high order simplices or orthotopes. `http://www.math.univ-paris13.fr/~cuvelier/software/`, 2019. User's Guide.

Informations for git maintainers of the simesh Matlab toolbox

git informations on the toolboxes used to build this manual

```
------------------------------------------------
  name : fc-simesh
   tag : 0.4.7
commit : be81202146bc0f1624b9fea86b89b09b56d497ff
  date : 2025-01-29
  time : 09-49-19
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 : 795429326c1c20a60c186f27df2fefd24680a431
  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-siplt
   tag : 0.2.7
commit : 76eee19adfdaefe94212bc032dd43f186a7e3b7e
  date : 2025-01-29
  time : 13-16-15
status : 0
------------------------------------------------
```

git informations on the LaTeX package used to build this manual

```
------------------------------------------------
  name : fctools
   tag :
commit : 03d38737a795cdbf4e1a8754470e963cdfe83316
  date : 2025-01-24
  time : 09:58:52
status : 1
------------------------------------------------
```

Using the remote configuration repository:

url	ssh://lagagit/MCS/Cuvelier/Matlab/fc-config
commit	0aad3be99ddb49fc11f3877f9c97f38e4c1885ba