User's Guide *

François Cuvelier†

December 13, 2017

Abstract

This object-oriented Octave packageallows to use simplices meshes generated from `gmsh` (in dimension 2 or 3) or an hypercube triangulation (in any dimension). For graphical representation the FC-SIPLT package is used.

0 Contents

*Compiled with Octave 4.2.1

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

This work was supported by the ANR project DEDALES under grant ANR-14-CE23-0005.

1 Introduction

This package was tested under

Windows 10.0.16299: with Octave 4.2.0 and 4.2.1

Mac OS X 10.12.6: with Octave 4.2.1 (installed with homebrew)

Ubuntu 14.04.5 LTS: with Octave 4.2.0 and 4.2.1 (both compiled from source)

Ubuntu 16.04.3 LTS: with Octave 4.2.0 and 4.2.1 (both compiled from source)

Ubuntu 17.10: with Octave 4.2.0 and 4.2.1 (both compiled from source)

It is not compatible with Octave 4.0.x and previous.

2 Installation

2.1 Installation automatic, all in one (recommanded)

For this method, one just have to get/download the install file

ofc_simesh_install.m

or get it on the dedicated web page. Thereafter, one run it under Octave. This command download, extract and configure the *fc-simesh* and the required packages (*fc-tools*, *fc-oogmsh*, *fc-hypermesh*) in the current directory.

For example, to install this package in `~/Octave/packages` directory, one have to copy the file ofc_simesh_install.m in the `~/Octave/packages` directory. Then in a Octave terminal run the following commands to install the *fc-simesh* package with graphical extension

```
>> cd ~/Octave/packages
>> ofc_simesh_install
```

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

```
Parts of the GNU Octave <fc-simesh> package.
Copyright (C) 2016-2017 Francois Cuvelier <cuvelier@math.univ-paris13.fr>

1- Downloading and extracting the packages
   -> <fc-tools>[0.0.19] ... OK
   -> <fc-hypermesh>[0.0.6] ... OK
   -> <fc-oogmsh>[0.0.17] ... OK
   -> <fc-simesh>[0.2.1] ... OK
   -> <fc-graphics4mesh>[0.0.2] ... OK
   -> <fc-siplt>[0.0.2] ... OK
2- Setting the packages
2-a) Setting the <fc-hypermesh> package
Write in ...
     ~/Octave/packages/fc-simesh-full/fc_hypermesh-0.0.6/configure_loc.m ...
     ...
  -> done
2-b) Setting the <fc-oogmsh> package
[fc-oogmsh] Using GMSH binary : ~/bin/gmsh
[fc-oogmsh] Writing in ...
     ~/Octave/packages/fc-simesh-full/fc_oogmsh-0.0.17/configure_loc.m ...
[fc-oogmsh] configured with
   ->      gmsh_bin='~/bin/gmsh';
   ->      ...
        mesh_dir='~/Octave/packages/fc-simesh-full/fc_oogmsh-0.0.17/meshes';
   ->       ...
        geo_dir='~/Octave/packages/fc-simesh-full/fc_oogmsh-0.0.17/geodir';
   -> fc_tools_dir='~/Octave/packages/./fc-simesh-full/fc_tools-0.0.19';
[fc-oogmsh] done
2-c) Setting the <fc-simesh> package without graphics
[fc-simesh] Unable to load the fc-siplt toolbox/package in current path
[fc-simesh] Guess path does not exists:
   -> siplt
[fc-] Guess path does not exists:
   -> [fc-simesh] Use fc_simesh.configure('fc_siplt_dir',<DIR>) to ...
        correct this issue

[fc-simesh] no graphics package installed
[fc-simesh] Writing in ...
     ~/Octave/packages/fc-simesh-full/fc_simesh-0.2.1/configure_loc.m ...
[fc-simesh] configured with
   -> oogmsh_dir    ...
        ='~/Octave/packages/./fc-simesh-full/fc_oogmsh-0.0.17';
   -> hypermesh_dir ...
        ='~/Octave/packages/./fc-simesh-full/fc_hypermesh-0.0.6';
   -> siplt_dir     ='';
[fc-simesh] done
2-d) Setting the <fc-graphics4mesh> toolbox
Write in ...
     ~/Octave/packages/fc-simesh-full/fc_graphics4mesh-0.0.2/configure_loc.m ...
     ...
  -> done
2-e) Setting the <fc-siplt> toolbox
Write in ...
     ~/Octave/packages/fc-simesh-full/fc_siplt-0.0.2/configure_loc.m ...
  -> done
2-f) Setting the <fc-simesh> toolbox with graphics
[fc-simesh] Writing in ...
     ~/Octave/packages/fc-simesh-full/fc_simesh-0.2.1/configure_loc.m ...
[fc-simesh] configured with
   -> oogmsh_dir    ...
        ='~/Octave/packages/./fc-simesh-full/fc_oogmsh-0.0.17';
   -> hypermesh_dir ...
        ='~/Octave/packages/./fc-simesh-full/fc_hypermesh-0.0.6';
   -> siplt_dir     ='~/Octave/packages/./fc-simesh-full/fc_siplt-0.0.2';
[fc-simesh] done
3- Using instructions
   To use the <fc-simesh> package:
   addpath('~/Octave/packages/./fc-simesh-full/fc_simesh-0.2.1')
   fc_simesh.init()

   See ~/Octave/packages/ofc_simesh_set.m
```

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

directory `~/Octave/packages/fc-simesh-full` and, for each Octave session, one have to set the package by:

```
>> addpath('~/Octave/packages/fc-simesh-full/fc-simesh-0.2.1')
>> fc_simesh.init()
```

To install the *fc-simesh* package without graphical extension one can use the following command

```
>> ofc_simesh_install('graphics',false)
```

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

`~/Octave/packages/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 $\mathbb{R}^{\dim}$, $k \leqslant \dim$, is a polytope which is the convex hull of its $k+1$ vertices of $\mathbb{R}^{\dim}$. More formally, suppose the $k+1$ vertices $\mathrm{q}^0, \cdots, q^k \in \mathbb{R}^{\dim}$ 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 $\mathbb{R}^{\dim}$, $k \leqslant \dim$, a mesh with **unique label** only composed with k-simplices.

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

3.1 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 ($0 \leqslant d \leqslant dim$)
n_q	:	integer number of vertices
n_{me}	:	integer number of elements (d-simplices)
q	:	dim-by-n_q array of reals array of vertex coordinates
me	:	$(d+1)$-by-n_{me} array of integers connectivity array for **mesh elements**
vols	:	1-by-n_{me} array of reals array of mesh element volumes
h	:	double mesh step size (=maximum edge length in the mesh)
`toGlobal`	:	1-by-n_q 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-n_q 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, `nqParents{2}` indices array which convert local vertices numbering to the *parent* of the *parent* mesh, `nqParents{end}` indices array which convert local vertices numbering to the global mesh.

More precisely

- $q(\nu, j)$ is the ν-th coordinate of the j-th vertex, $\nu \in \{1, \ldots, dim\}$, $j \in \{1, \ldots, n_q\}$. The j-th vertex will be also denoted by $q^j = q(:, j)$.
- $me(\beta, k)$ is the storage index of the β-th vertex of the k-th element (d-simplex), in the array q, for $\beta \in \{1, ..., d+1\}$ and $k \in \{1, \ldots, n_{me}\}$. So $q(:, me(\beta, k))$ represents the coordinates of the β-th vertex of the k-th mesh element.

- vols(k) is the volume of the k-th d-simplex .

3.2 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 ($l \leqslant$ 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
`sThlab`	:	array of `nsTh` integers in `sTh` label of i-th SIMESHELT object in `sTh` is number `sThlab`(i)
n_q	:	integer number of vertices in the mesh
`toGlobal`	:	1-by-n_q 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-n_q 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 Only used with partitioned mesh and the FC-PSIMESH package.
`toParents`	:	1-by-n cell array Only used with partitioned mesh and the FC-PSIMESH package.

Let $\mathcal{T}_h$ be a SIMESH object. The global dim-by-$\mathcal{T}_h$.n_q array q of mesh vertices is not explicitly stored in $\mathcal{T}_h$, 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

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

Listing 1: : 2D siMesh object from sample20.geo

```
meshfile=gmsh.buildmesh2d('sample20',20,'force',false);
Th=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] Starting building mesh ...
    /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/sample20-20.msh with gmsh 3.0.6
[fc-oogmsh] Using command : gmsh -2 -setnumber N 20 <fc-oogmsh>/geodir/2d/sample20.geo -o ...
    /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/sample20-20.msh
 Be patient...
*** Th:
  siMesh with properties:
          bbox: [ -1 1 -1 1 ] (1x4 double)
             d: 2 double
           dim: 2 double
            nq: 2558 double
     nqParents: 2558 double
          nsTh: 11 double
           sTh: (1x11 cell)
     sThcolors: (11x3 double)
     sThgeolab: []
        sThlab: [ 1 2 20 101 102 103 104 1 2 10 20 ] (1x11 double)
   sThpartlabs: []
    sThphyslab: [ 1 2 10 20 ] (1x4 double)
       sThsimp: [ 1 1 1 1 1 1 1 2 2 2 2 ] (1x11 double)
      toGlobal: (1x2558 double)
      toParent: (1x2558 double)
     toParents: (1x1 cell)
*** Th.sTh{9}:
  siMeshElt with properties:
         Tag: (1x28 char)
        bbox: [ 0.4 0.6 -0.1 0.1 ] (1x4 double)
       color: [ 1 0 0 ] (1x3 double)
           d: 2 double
         dim: 2 double
      geolab: (60x1 double)
    gradBaCo: (60x3 double)
           h: 0.051428 double
       label: 2 double
          me: (3x60 double)
         nme: 60 double
          nq: 39 double
    nqGlobal: 2558 double
    nqParent: 2558 double
   nqParents: 2558 double
     partlab: []
           q: (2x39 double)
    toGlobal: (1x39 double)
    toParent: (1x39 double)
   toParents: (1x1 cell)
        vols: (1x60 double)
[fc-tools] waiting 2(s) to finish saving figures
```

From the output of the Listing 1 or from the Figure 1 the complete domain is

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

and we note

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

So this mesh is 2-simplicial mesh in $\mathbb{R}^2$ and is composed of :

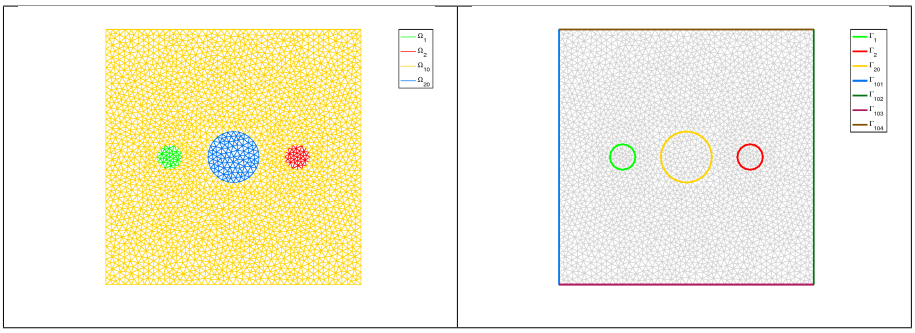

Figure 1: 2D siMesh object from `sample20.geo`

- four 2-simplicial elementary meshes : Ω_i, $\forall i \in \{1, 2, 10, 20\}$
- seven 1-simplicial elementary meshes : Γ_i $\forall i \in \{1, 2, 20, 101, 102, 104\}$

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

Listing 2: : 3D Mesh from `quart_sphere2.geo`

```
meshfile=gmsh.buildmesh3d('quart_sphere2',5);
Th=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] Starting building mesh ...
    /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/quart_sphere2-5.msh with gmsh ...
    3.0.6
[fc-oogmsh] Using command : gmsh -3 -setnumber N 5 <fc-oogmsh>/geodir/3d/quart_sphere2.geo -o ...
    /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/quart_sphere2-5.msh
 Be patient...
  Mesh /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/quart_sphere2-5.msh is a ...
      3-dimensional mesh
   Force dimension to 3
*** Th:
  siMesh with properties:
          bbox: [ -1 1 0 1 0 1 ] (1x6 double)
             d: 3 double
           dim: 3 double
            nq: 1228 double
     nqParents: 1228 double
          nsTh: 23 double
           sTh: (1x23 cell)
     sThcolors: (23x3 double)
     sThgeolab: []
        sThlab: [ 1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 1 2 1 2 3 4 5 ] (1x23 double)
   sThpartlabs: []
    sThphyslab: [ 1 2 ] (1x2 double)
       sThsimp: [ 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 3 3 0 0 0 0 0 ] (1x23 double)
      toGlobal: (1x1228 double)
      toParent: (1x1228 double)
     toParents: (1x1 cell)
*** Th.sTh{9}:
  siMeshElt with properties:
         Tag: (1x28 char)
        bbox: (1x6 double)
       color: [ 0 0 1 ] (1x3 double)
           d: 1 double
         dim: 3 double
      geolab: (15x1 double)
    gradBaCo: (15x2 double)
           h: 0.104672 double
       label: 9 double
          me: (2x15 double)
         nme: 15 double
          nq: 16 double
    nqGlobal: 1228 double
    nqParent: 1228 double
   nqParents: 1228 double
     partlab: []
           q: (3x16 double)
    toGlobal: (1x16 double)
    toParent: (1x16 double)
   toParents: (1x1 cell)
        vols: (1x15 double)
[fc-siplt] 'FaceAlpha' or 'EdgeAlpha' not yet implemented in plotmesh
[fc-tools] waiting 2(s) to finish saving figures
[fc-tools] waiting 2(s) to finish saving figures
```

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

- two 3-simplicial elementary meshes : Ω_i, $\forall i \in \{1, 2\}$
- seven 2-simplicial elementary meshes : Γ_i $\forall i \in [\![1, 7]\!]$
- nine 1-simplicial elementary meshes : $\partial\Gamma_i$ $\forall i \in [\![1, 9]\!]$

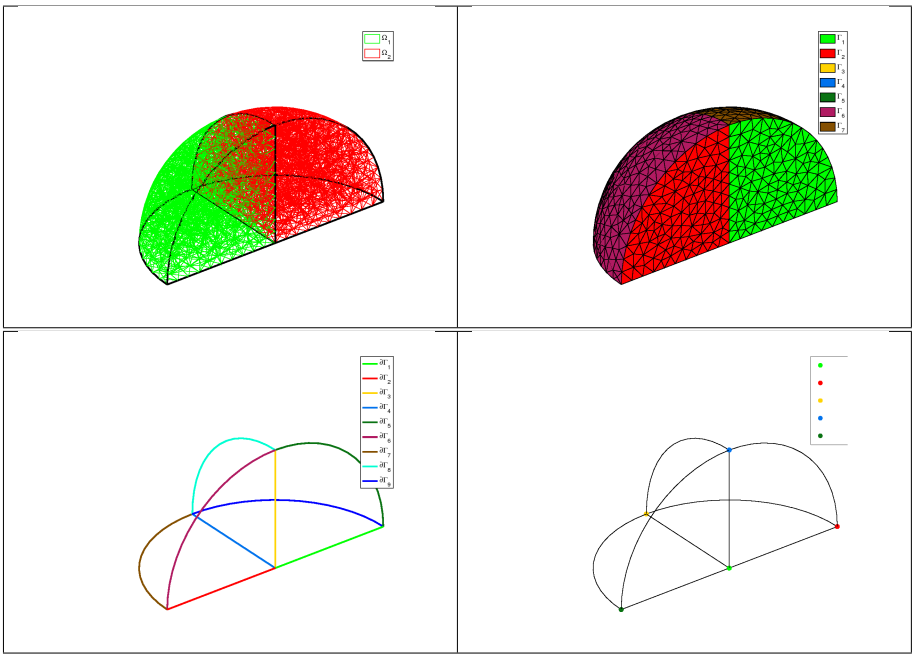

Figure 2: 3D Mesh from `quart_sphere2.geo`

- five 0-simplicial elementary meshes : $\partial^2\Gamma_i \ \forall i \in [\![1,5]\!]$

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

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

```
meshfile=gmsh.buildmesh3ds('demisphere4surf',5,'force',true);
Th=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] Starting building mesh ...
    /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/demisphere4surf-5.msh with ...
    gmsh 3.0.6
[fc-oogmsh] Using command : gmsh -2 -setnumber N 5 <fc-oogmsh>/geodir/3ds/demisphere4surf.geo ...
    -o /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/demisphere4surf-5.msh
 Be patient...
  Mesh /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/demisphere4surf-5.msh is a ...
      3-dimensional mesh
   Force dimension to 3
*** Th:
  siMesh with properties:
          bbox: [ -1 1 -1 1 0 1 ] (1x6 double)
             d: 2 double
           dim: 3 double
            nq: 247 double
     nqParents: 247 double
          nsTh: 12 double
           sTh: (1x12 cell)
     sThcolors: (12x3 double)
     sThgeolab: []
        sThlab: [ 1 2 3 4 5 6 7 8 1 2 3 4 ] (1x12 double)
   sThpartlabs: []
    sThphyslab: [ 1 2 3 4 ] (1x4 double)
       sThsimp: [ 1 1 1 1 1 1 1 1 2 2 2 2 ] (1x12 double)
      toGlobal: (1x247 double)
      toParent: (1x247 double)
     toParents: (1x1 cell)
*** Th.sTh{9}:
  siMeshElt with properties:
         Tag: (1x29 char)
        bbox: (1x6 double)
       color: [ 0 1 0 ] (1x3 double)
           d: 2 double
         dim: 3 double
      geolab: (114x1 double)
    gradBaCo: (114x3 double)
           h: 0.272289 double
       label: 1 double
          me: (3x114 double)
         nme: 114 double
          nq: 70 double
    nqGlobal: 247 double
    nqParent: 247 double
   nqParents: 247 double
     partlab: []
           q: (3x70 double)
    toGlobal: (1x70 double)
    toParent: (1x70 double)
   toParents: (1x1 cell)
        vols: (1x114 double)
[fc-siplt] 'FaceAlpha' or 'EdgeAlpha' not yet implemented in plotmesh
[fc-tools] waiting 2(s) to finish saving figures
```

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

- four 2-simplicial elementary meshes : Ω_i, $\forall i \in [\![1, 4]\!]$
- eight 1-simplicial elementary meshes : Γ_i $\forall i \in [\![1, 8]\!]$

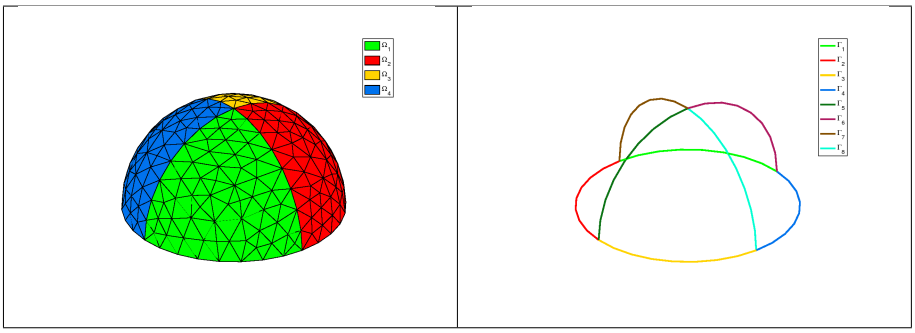

Figure 3: 3D surface Mesh from `demisphere4surf.geo`, label of the domains (left) and label of the boundaries (right)

3.4 siMesh object methods

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=siMesh(meshfile)

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

Description

`Th=siMesh(meshfile)` create the siMesh object Th from the mesh file meshfile (`gmsh` format by default).

Th=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 gmsh.buildmesh2d of the fc-oogmsh package to build the mesh from the *.geo* file `condenser11.geo`. This .geo file is located in the directory `geodir/2d` of the fc-oogmsh package.

Matlab commands with output

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

```
*** Read mesh ***
Th =

  siMesh with properties:
          bbox: [ -1 1 -1 1 ] (1x4 double)
             d: 2 double
           dim: 2 double
            nq: 3474 double
     nqParents: 3474 double
          nsTh: 19 double
           sTh: (1x19 cell)
     sThcolors: (19x3 double)
     sThgeolab: []
        sThlab: [ 1 2 3 4 5 6 7 8 20 101 102 103 104 2 4 6 8 10 20 ] (1x19 double)
   sThpartlabs: []
    sThphyslab: [ 2 4 6 8 10 20 ] (1x6 double)
       sThsimp: [ 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 ] (1x19 double)
      toGlobal: (1x3474 double)
      toParent: (1x3474 double)
     toParents: (1x1 cell)
```

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 4: : siMesh find method samples

```
meshfile=fc_oogmsh.buildmesh3d('quart_sphere2',5, 'verbose',0);
Th=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

```
  siMesh with properties:
          bbox: [ -1 1 0 1 0 1 ] (1x6 double)
             d: 3 double
           dim: 3 double
            nq: 1228 double
     nqParents: 1228 double
          nsTh: 23 double
           sTh: (1x23 cell)
     sThcolors: (23x3 double)
     sThgeolab: []
        sThlab: [ 1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 1 2 1 2 3 4 5 ] (1x23 double)
    sThpartlabs: []
     sThphyslab: [ 1 2 ] (1x2 double)
        sThsimp: [ 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 3 3 0 0 0 0 0 ] (1x23 double)
      toGlobal: (1x1228 double)
      toParent: (1x1228 double)
     toParents: (1x1 cell)
3-simplices siMeshElt
  indices: [17 18], labels=[1 2]
2-simplices siMeshElt
  indices: [10 11 12 13 14 15 16], labels=[1 2 3 4 5 6 7]
2-simplices siMeshElt with label==4
  indices: [13], labels=[4]
2-simplices siMeshElt with label in [6,4,2,10]
  indices: [11 13 15], labels=[2 4 6]
```

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(), 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 5 four approaches to defined this function for using with **feval** method.

Listing 5: : **feval** method, four ways to defined a function

```
meshfile=fc_oogmsh.buildmesh2d('condenser11',50,'verbose',0);
Th=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 6: : **feval** method with a vector-valued function

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

  siMesh with properties:
          bbox: [ -1 1 -1 1 ] (1x4 double)
             d: 2 double
           dim: 2 double
            nq: 13258 double
     nqParents: 13258 double
          nsTh: 19 double
           sTh: (1x19 cell)
     sThcolors: (19x3 double)
     sThgeolab: []
        sThlab: [ 1 2 3 4 5 6 7 8 20 101 102 103 104 2 4 6 8 10 20 ] (1x19 double)
   sThpartlabs: []
    sThphyslab: [ 2 4 6 8 10 20 ] (1x6 double)
       sThsimp: [ 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 ] (1x19 double)
      toGlobal: (1x13258 double)
      toParent: (1x13258 double)
     toParents: (1x1 cell)

*** nq=13258, size(z)==[13258,3]
```

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 7: : **eval** method, four ways to defined a function

```
meshfile=fc_oogmsh.buildmesh2d('condenser11',50,'verbose',0);
Th=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)=[13258,1]
size(z2)=[13258,1]
size(z3)=[13258,1]
max(abs(z2-z1))=0.000000e+00
max(abs(z3-z1))=0.000000e+00
```

Sample 2

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

```
meshfile=fc_oogmsh.buildmesh2d('condenser11',50,'verbose',0);
Th=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=13258, size(z)==[13258,4]
```

get_h method

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

- h=Th.get_h()

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 9: : get_mesh method, four ways to defined a function

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

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 10: : get_nme method

```
meshfile=gmsh.buildmesh3d('quart_sphere2',5);
Th=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 ...
    /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/quart_sphere2-5.msh already ...
    exists.
  -> Use "force" flag to rebuild if needed.
  Mesh /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/quart_sphere2-5.msh is a ...
      3-dimensional mesh
   Force dimension to 3
Number of 3-simplices : 4874
Number of 2-simplices : 1803
Number of 1-simplices : 115
Number of 0-simplices : 5
Number of 2-simplices in union of label's 1 to 4 : 788
```

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 11: : get_nqe method

```
meshfile=gmsh.buildmesh3d('quart_sphere2',5);
Th=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 ...
    /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/quart_sphere2-5.msh already ...
    exists.
  -> Use "force" flag to rebuild if needed.
  Mesh /tmp/tmp.Q6UcvHiF7L/fc-simesh-full/fc_oogmsh-0.0.17/meshes/quart_sphere2-5.msh is a ...
      3-dimensional mesh
   Force dimension to 3
Number of vertices in 3-simplices elementary meshes : 1228
Number of vertices in 2-simplices elementary meshes : 887
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 : 425
```

3.5 Hypercube as a SIMESH object

The function fc_simesh.HyperCube allows to create a SIMESH object representing an hypercube in any dimension. It uses the FC-HYPERMESH Octave package.

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

2D hypercube

In Listing 12 a usage example generating a 2D hypercube as a SIMESH object is given. This SIMESH object is representing in Figure 4 by using the FC-SIPLT package.

Listing 12: : 2D Hypercube SIMESH object generated with the function siMesh.HyperCube

```
Th=fc_simesh.HyperCube(2,10);
disp(Th)
```

Output

```
  siMesh with properties:
          bbox: [ 0 1 0 1 ] (1x4 double)
             d: 2 double
           dim: 2 double
            nq: 121 double
     nqParents: []
          nsTh: 9 double
           sTh: (1x9 cell)
     sThcolors: (9x3 double)
     sThgeolab: []
        sThlab: [ 1 1 2 3 4 1 2 3 4 ] (1x9 double)
   sThpartlabs: []
    sThphyslab: 1 double
       sThsimp: [ 2 1 1 1 1 0 0 0 0 ] (1x9 double)
      toGlobal: (1x121 double)
      toParent: []
     toParents: []
[fc-tools] waiting 2(s) to finish saving figures
```

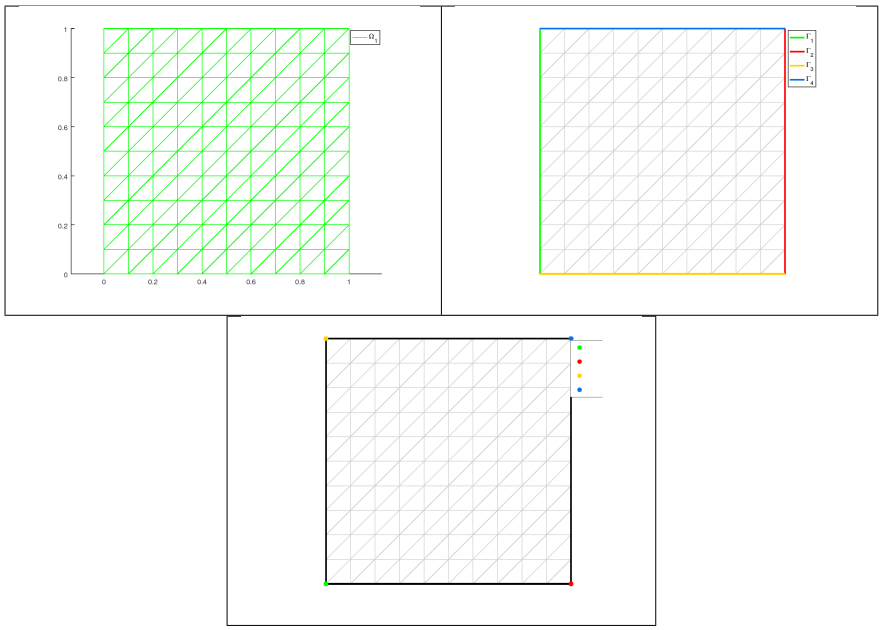

Figure 4: 2D Hypercube SIMESH object generated with the function fc_simesh.HyperCube, representation of the elementary meshes with 2-simplices (top left), 1-simplices (top right) and 0-simplices (bottom)

3D hypercube

In Listing 13 a usage example generating a 3D hypercube as a SIMESH object is given. This SIMESH object is representing in Figure 5 by using the the FC-SIPLT package. .

Listing 13: : 3D Hypercube SIMESH object generated with the function fc_simesh.HyperCube

```
Th=fc_simesh.HyperCube(3,10);
disp(Th)
```

Output

```
  siMesh with properties:
          bbox: [ 0 1 0 1 0 1 ] (1x6 double)
             d: 3 double
           dim: 3 double
            nq: 1331 double
     nqParents: []
          nsTh: 27 double
           sTh: (1x27 cell)
     sThcolors: (27x3 double)
     sThgeolab: []
        sThlab: (1x27 double)
   sThpartlabs: []
    sThphyslab: 1 double
       sThsimp: (1x27 double)
      toGlobal: (1x1331 double)
      toParent: []
     toParents: []
[fc-tools] waiting 2(s) to finish saving figures
```

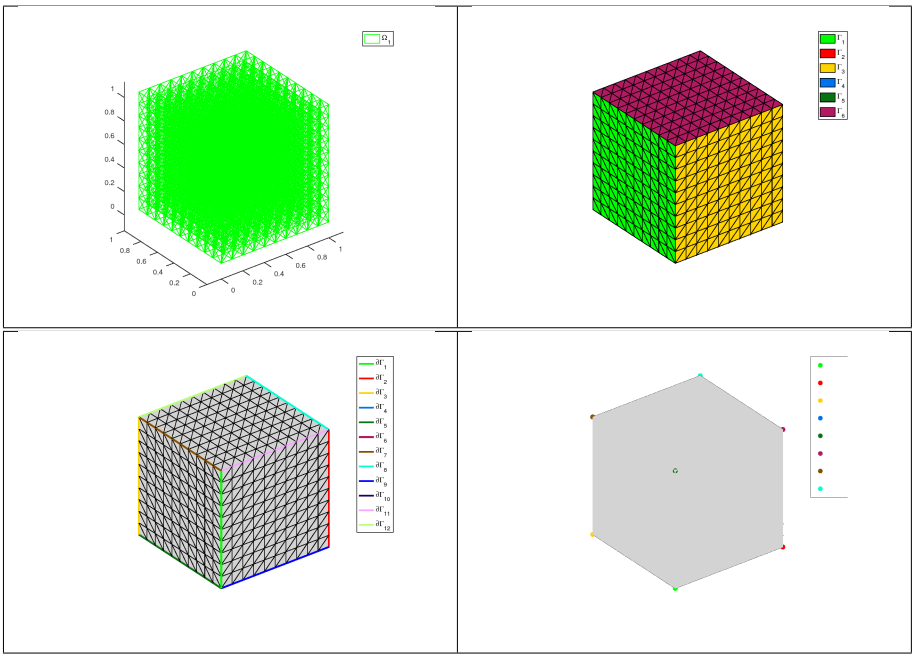

Figure 5: 3D Hypercube siMesh object generated with the function siMesh.HyperCube, representation of the elementary meshes with 3-simplices (top left), 2-simplices (top right), 1-simplices (bottom left) and 0-simplices (bottom right)

4D hypercube

In Listing 14 a usage example generating a 4D hypercube as a siMesh object is given.

Listing 14: : function fc_simesh.HyperCube

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

Output

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

5D hypercube

In Listing 14 a usage example generating a 5D hypercube as a siMesh object is given.

Listing 15: : function siMesh.HyperCube

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

Output

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