Documentation of the oogmsh Octave package version 0.0.19[*]

François Cuvelier[†]

September 26, 2018

Abstract

This experimental Octave package make it possible to generate mesh files from *.geo* files by using `gmsh`. It's also possible with the ooGmsh class to read the mesh file and to store its contains in more user-friendly form. This toolbox must be regarded as a very simple interface between gmsh files and Octave . So you are free to create any data structures or objects you want from an ooGmsh object.

0 Contents

[*]Compiled with Octave 4.2.1 under hercule computer

[†]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

The oogmsh Octave package is closely related to **gmsh**, see [2] or [1], which is a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. **gmsh** can also build two-dimensional meshes and three-dimensional surface meshes. This package was initialy created to make it possible from Octave to rapidly

- generate mesh file from .geo file by using **gmsh**
- efficiently read this mesh file and store its contents in ooGmsh Octave object easy to manipulate.

The ooGmsh Octave object can be used to create, from a .msh file, any data structures or objects needed by your project. For example, the fc-simesh Octave package uses this package to create the siMesh object containing all the simplices elements of the mesh.

This package was tested under

OS	**Octave**	**gmsh**
Ubuntu 18.04	4.2.0 to 4.4.1	3.0.x, with x from 0 to 6 4.0.x, with x from 0 to 1
macOS High Sierra 10.13.6	4.2.2	3.0.x, with x from 0 to 6 4.0.x, with x from 0 to 1
Windows 10 (1803)	4.2.0 to 4.2.1	3.0.x, with x from 0 to 6 4.0.x, with x from 0 to 1

Firstly, we explain how to configure the oogmsh package for using **gmsh**. Thereafter, we describe the oogmsh 's functions which use **gmsh** to create mesh files.

2 Installation

2.1 Installation automatic, all in one (recommanded)

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

ofc_oogmsh_install.m

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

By default, the gmsh binary is supposed to be located in

- `<USERDIR>/bin/gmsh` under linux,

- `<USERDIR>/GMSH/Gmsh.app/Contents/MacOS/gmsh` under Mac OS X,
- `<USERDIR>/Softwares/GMSH/gmsh.exe` (32 bit version) under Windows

It can be directly given by using the `'gmsh_bin'` option of the install command:

```
ofc_oogmsh_install('gmsh_bin', '<GMSH>')
```

where `<GMSH>` is the gmsh binary with path.
It's also possible, after installation, to change the gmsh binary by using

```
fc_oogmsh.configure('gmsh_bin','<GMSH>')
```

command under Octave.

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

```
>> cd ~/Octave/packages
>> ofc_oogmsh_install
```

There is the output of the `ofc_oogmsh_install` command:

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

1- Downloading and extracting the packages
2- Setting the <fc-oogmsh> package
[fc-oogmsh] Writing in ~/Octave/packages/fc-oogmsh-full/fc_oogmsh-0.0.19/
     configure_loc.m ...
3- Using packages:
   ->             fc-tools : 0.0.23
   ->            fc-oogmsh : 0.0.19
*** Using instructions
   To use the <fc-oogmsh> package:
   addpath('~/Octave/packages/fc-oogmsh-full/fc_oogmsh-0.0.19')
   fc_oogmsh.init()

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

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

```
>> addpath('~/Octave/packages/fc-oogmsh-full/fc-oogmsh-0.0.19')
>> fc_oogmsh.init()
```

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

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

3 gmsh interface

All functions provided in this section use `gmsh` to create a mesh file from a `gmsh` geometry script file (extension *.geo*).

3.1 function gmsh.buildmesh2d

This function uses `gmsh` and a *.geo* file (describing a 2D-geometry) to generate a 2D-mesh.

Syntaxe

```
meshfile=gmsh.buildmesh2d(geofile,N)

meshfile=gmsh.buildmesh2d(geofile,N,Name,Value)
```

Description

$\boxed{\text{meshfile=gmsh.buildmesh2d(geofile,N)}}$ create a 2D-mesh using `gmsh` and the *geo* file geofile (without path). The integer N has two functions : numbering the name of the generated mesh as <geofile without extension and path> + <-N.msh> and passing this number to `gmsh` via the option "-setnumber N <N>". Usually we used this parameter in `gmsh` to set the prescribed mesh element size at the points. (see given *geo* files)
As output return a file name (with full path) corresponding to the mesh generated by `gmsh`.

meshfile=gmsh.buildmesh2d(geofile,N,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. The Name options can be

- `'geodir'` : to specify the directory of the *geo* file geofile ,
- `'meshdir'` : to specify the directory where the mesh file will be written,
- `'meshfile'` : to specify the name of the mesh file (with path and `.msh` extension),
- `'check'` : to perform various consistency checks on mesh with `gmsh`, if Value is true. (default : false)
- `'force'` : to force meshing even if the mesh file already exists if Value is true (default : false)
- `'verbose'` : to specify the degree of verbosity (0, silence; 2, default; ...)
- `'strings'` : cells array of strings corresponding to `gmsh` options given with `-string "..."` (default empty) (see `gmsh` documentation)

Examples All the following examples use the *.geo* file `condenser11.geo` which is in the directory `geodir` of the toolbox.

Octave commands with output

```
disp('****␣gmsh.buildmesh2d␣:␣1st␣call')
meshfile=gmsh.buildmesh2d('condenser11',25,'force',true);
disp('****␣gmsh.buildmesh2d␣:␣2nd␣call')
meshfile=gmsh.buildmesh2d('condenser11',25);
```

```
**** gmsh.buildmesh2d : 1st call
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/2d/condenser11.geo
[fc-oogmsh] Starting building mesh <fc-oogmsh>/meshes/condenser11-25.msh with gmsh 4.0.1
[fc-oogmsh] Using command : gmsh -2 -setnumber N 25 -string "Mesh.MshFileVersion=2;" <fc-oogmsh>/geodir/2d/condenser11.geo -o ...
       <fc-oogmsh>/meshes/condenser11-25.msh
 Be patient...
**** gmsh.buildmesh2d : 2nd call
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/2d/condenser11.geo
[fc-oogmsh] Mesh file <fc-oogmsh>/meshes/condenser11-25.msh already exists.
  -> Use "force" flag to rebuild if needed.
```

Octave commands with output

```
meshfile=gmsh.buildmesh2d('condenser11',25,'force',true, ...
    'verbose',4, 'strings',{'Mesh.Algorithm=1;', ...
    'Mesh.ScalingFactor=2;'});
```

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/2d/condenser11.geo
[fc-oogmsh] Overwritting mesh file <fc-oogmsh>/meshes/condenser11-25.msh
[fc-oogmsh] Starting building mesh <fc-oogmsh>/meshes/condenser11-25.msh with gmsh 4.0.1
[fc-oogmsh] Using command : gmsh -2 -setnumber N 25 -string "Mesh.Algorithm=1;Mesh.ScalingFactor=2;Mesh.MshFileVersion=2;" ...
       <fc-oogmsh>/geodir/2d/condenser11.geo -o <fc-oogmsh>/meshes/condenser11-25.msh
 Be patient...
[fc-oogmsh] gmsh output :
Info    : Running '/home/cuvelier/bin/gmsh -2 -setnumber N 25 -string Mesh.Algorithm=1;Mesh.ScalingFactor=2;Mesh.MshFileVersion=2; ...
       <fc-oogmsh>/geodir/2d/condenser11.geo -o <fc-oogmsh>/meshes/condenser11-25.msh' [Gmsh 4.0.1, 1 node, max. 1 thread]
Info    : Started on Wed Sep 26 08:30:04 2018
Info    : Reading '<fc-oogmsh>/geodir/2d/condenser11.geo'...
Info    : Reading '<fc-oogmsh>/geodir/2d/options01_data.geo'...
Info    : Done reading '<fc-oogmsh>/geodir/2d/options01_data.geo'
Info    : Reading '<fc-oogmsh>/geodir/2d/shape_functions.geo'...
Info    : Done reading '<fc-oogmsh>/geodir/2d/shape_functions.geo'
Info    : Removing duplicate mesh vertices...
Info    : Found 0 duplicate vertices
Info    : No duplicate vertices found
Info    : Done reading '<fc-oogmsh>/geodir/2d/condenser11.geo'
Info    : Meshing 1D...
Info    : Meshing curve 101 (Line)
Info    : Meshing curve 102 (Line)
Info    : Meshing curve 103 (Line)
Info    : Meshing curve 104 (Line)
Info    : Meshing curve 106 (Circle)
Info    : Meshing curve 107 (Circle)
Info    : Meshing curve 108 (Circle)
Info    : Meshing curve 109 (Circle)
Info    : Meshing curve 111 (Circle)
Info    : Meshing curve 112 (Circle)
Info    : Meshing curve 113 (Circle)
Info    : Meshing curve 114 (Circle)
Info    : Meshing curve 116 (Circle)
Info    : Meshing curve 117 (Circle)
Info    : Meshing curve 118 (Circle)
Info    : Meshing curve 119 (Circle)
Info    : Meshing curve 121 (Circle)
Info    : Meshing curve 122 (Circle)
Info    : Meshing curve 123 (Circle)
Info    : Meshing curve 124 (Circle)
Info    : Meshing curve 126 (Circle)
Info    : Meshing curve 127 (Circle)
Info    : Meshing curve 128 (Circle)
Info    : Meshing curve 129 (Circle)
Info    : Meshing curve 131 (Circle)
Info    : Meshing curve 132 (Circle)
Info    : Meshing curve 133 (Circle)
Info    : Meshing curve 134 (Circle)
Info    : Meshing curve 136 (Circle)
Info    : Meshing curve 137 (Circle)
Info    : Meshing curve 138 (Circle)
Info    : Meshing curve 139 (Circle)
Info    : Meshing curve 141 (Circle)
Info    : Meshing curve 142 (Circle)
Info    : Meshing curve 143 (Circle)
Info    : Meshing curve 144 (Circle)
Info    : Meshing curve 146 (Circle)
Info    : Meshing curve 147 (Circle)
Info    : Meshing curve 148 (Circle)
Info    : Meshing curve 149 (Circle)
Info    : Done meshing 1D (0.006434 s)
Info    : Meshing 2D...
Info    : Meshing surface 105 (Plane, MeshAdapt)
Info    : Meshing surface 110 (Plane, MeshAdapt)
Info    : Meshing surface 120 (Plane, MeshAdapt)
Info    : Meshing surface 130 (Plane, MeshAdapt)
Info    : Meshing surface 140 (Plane, MeshAdapt)
Info    : Meshing surface 150 (Plane, MeshAdapt)
Info    : Done meshing 2D (0.253911 s)
Info    : 3092 vertices 6317 elements
Info    : Writing '<fc-oogmsh>/meshes/condenser11-25.msh'...
Info    : Done writing '<fc-oogmsh>/meshes/condenser11-25.msh'
Info    : Stopped on Wed Sep 26 08:30:04 2018
```

3.2 function gmsh.buildmesh3d

This function uses `gmsh` and a *.geo* file (describing a 3D-geometry) to generate a 3D-mesh. See function `gmsh.buildmesh2d` for usage and options.

3.3 function gmsh.buildmesh3ds

This function uses `gmsh` and a *.geo* file (describing a 3D surface geometry or a 3D-geometry) to generate a 3D surface mesh. See function `gmsh.buildmesh2d` for usage and options.

3.4 function gmsh.buildpartmesh2d

This function uses `gmsh` and a *.msh* file (containing a 2D-mesh) to generate a 2D partioned mesh.

Syntaxe

```
partmeshfile=gmsh.buildpartmesh2d(meshfile,np)

partmeshfile=gmsh.buildpartmesh2d(meshfile,np,Name,Value)
```

Description

partmeshfile=gmsh.buildpartmesh2d(meshfile,np) create a 2D partitioned mesh using `gmsh` and the *msh* file meshfile (with path). The integer np is the number of partitions.
As output return a file name (with full path) corresponding to the partitioned mesh generated by `gmsh`. The output file name is construct as following : <meshfile without extension>-part<np>.msh

partmeshfile=gmsh.buildpartmesh2d(meshfile,np,Name,Value, ...) specifies function options using one or more Name,Value pair arguments. The Name options can be

- `'savedir'` : to specify the directory where the partitioned mesh file will be written,
- `'check'` : to perform various consistency checks on mesh with `gmsh`, if Value is true. (default : false)
- `'force'` : to force meshing even if the mesh file already exists if Value is true (default : false)
- `'verbose'` : to specify the degree of verbosity (0, silence; 2, default; ...)
- `'strings'` : cells array of strings corresponding to `gmsh` options given with `-string "..."` (default empty) (see `gmsh` documentation)

Examples All the following examples use the `meshfile` as output of the command :
meshfile=gmsh.buildmesh2d('condenser11',25);

Octave commands with output

```
meshfile=gmsh.buildmesh2d('condenser11',25,'verbose',0);
pmfile=gmsh.buildpartmesh2d(meshfile,5,'force',true);
```

```
[fc-oogmsh] Input file : <fc-oogmsh>/meshes/condenser11-25.msh
[fc-oogmsh] Starting building mesh <fc-oogmsh>/meshes/condenser11-25-part5.msh with gmsh 4.0.1
[fc-oogmsh] Using command : gmsh -2 -saveall -part 5 -string "Mesh.MshFileVersion=2;" <fc-oogmsh>/meshes/condenser11-25.msh -o ...
        <fc-oogmsh>/meshes/condenser11-25-part5.msh
 Be patient...
```

Octave commands with output

```
meshfile=gmsh.buildmesh2d('condenser11',25,'verbose',0);
pmfile=gmsh.buildpartmesh2d(meshfile,5,'force',true,  'verbose',4, ...
    'strings',{'Mesh.MetisAlgorithm=3;'});
```

```
[fc-oogmsh] Input file : <fc-oogmsh>/meshes/condenser11-25.msh
[fc-oogmsh] Overwritting mesh file <fc-oogmsh>/meshes/condenser11-25-part5.msh
[fc-oogmsh] Starting building mesh <fc-oogmsh>/meshes/condenser11-25-part5.msh with gmsh 4.0.1
[fc-oogmsh] Using command : gmsh -2 -saveall -part 5 -string "Mesh.MetisAlgorithm=3;Mesh.MshFileVersion=2;" ...
        <fc-oogmsh>/meshes/condenser11-25.msh -o <fc-oogmsh>/meshes/condenser11-25-part5.msh
 Be patient...
[fc-oogmsh] gmsh output :
Info    : Running '/home/cuvelier/bin/gmsh -2 -saveall -part 5 -string Mesh.MetisAlgorithm=3;Mesh.MshFileVersion=2; ...
        <fc-oogmsh>/meshes/condenser11-25.msh -o <fc-oogmsh>/meshes/condenser11-25-part5.msh' [Gmsh 4.0.1, 1 node, max. 1 thread]
Info    : Started on Wed Sep 26 08:30:05 2018
Info    : Reading '<fc-oogmsh>/meshes/condenser11-25.msh'...
Info    : 3083 vertices
Info    : 6268 elements
Info    : Done reading '<fc-oogmsh>/meshes/condenser11-25.msh'
Info    : Meshing 1D...
Info    : Done meshing 1D (2.5e-05 s)
Info    : Meshing 2D...
Info    : Done meshing 2D (2e-05 s)
Info    : 3083 vertices 6268 elements
Info    : Partitioning mesh...
Info    : Running METIS graph partitioner
Info    : 5 partitions, 160 total edge-cuts
Info    : Done partitioning mesh (0.011817 s)
Info    : Creating partition topology...
Info    :  - Creating partition edges
Info    :  - Creating partition vertices
Info    : Done creating partition topology (0.002416 s)
Info    : Writing '<fc-oogmsh>/meshes/condenser11-25-part5.msh'...
Info    : Done writing '<fc-oogmsh>/meshes/condenser11-25-part5.msh'
Info    : Stopped on Wed Sep 26 08:30:05 2018
```

3.5 function gmsh.buildpartmesh3d

This function uses `gmsh` and a *.msh* file (containing of a 3D-mesh) to generate a 3D partioned mesh.

3.6 function gmsh.buildpartmesh3ds

This function uses `gmsh` and a *.msh* file (containing of a 3D surface mesh) to generate a 3D partioned surface mesh.

3.7 function gmsh.buildPartRectangle

This function uses `gmsh` and the *geodir/rectanglepart.geo* file to generate a 2D regular partioned mesh of the rectangle $[0, Lx] \times [0, Ly]$ with $Nx \times Ny$ partitions.

Syntaxe

```
meshfile=gmsh.buildpartrectangle(Lx,Ly,Nx,Ny,N)

meshfile=gmsh.buildpartrectangle(Lx,Ly,Nx,Ny,N, ...
    Name,Value)
```

Description

`meshfile=gmsh.buildpartrectangle(Lx,Ly,Nx,Ny,N)` create a 2D regular partitioned mesh using `gmsh` of the rectangle $[0, Lx] \times [0, Ly]$ with $Nx \times Ny$ partitions. The N parameter is passed to `gmsh` to set the prescribed mesh element size at the points
As output return a file name (with full path) corresponding to the partitioned mesh generated by `gmsh`. The default output file name is construct as following : rectanglepart-Lx%.3f-Ly%.3f-Nx%d-Ny%d-N%d.msh

`meshfile=gmsh.buildpartrectangle(Lx,Ly,Nx,Ny,N,Name,Value, ...)` specifies function options using one or more Name,Value pair arguments. The Name options can be

- `'meshdir'` : to specify the directory where the partitioned mesh file will be written,
- `'meshfile'` : to specify the mesh file name with .msh extension. Without path, the file is written in <meshdir> directory.
- `'check'` : to perform various consistency checks on mesh with `gmsh`, if Value is true. (default : false)
- `'force'` : to force meshing even if the mesh file already exists if Value is true (default : false)
- `'verbose'` : to specify the degree of verbosity (0, silence; 2, default; ...)
- `'strings'` : cells array of strings corresponding to `gmsh` options given with `-string "..."` (default empty) (see `gmsh` documentation)

Examples All the following examples ...

Octave commands with output

```
pmfile=gmsh.buildpartrectangle(1,1,3,2,100,'force',true);
```

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/2d/rectanglepart.geo
[fc-oogmsh] Starting building mesh <fc-oogmsh>/meshes/rectanglepart-Lx1.000-Ly1.000-Nx3-Ny2-N100.msh with gmsh 4.0.1
[fc-oogmsh] Using command : gmsh -2 -string "Mesh.MshFileVersion=2;" -setnumber N 100 -setnumber NX 3 -setnumber NY 2 -setnumber LX 1 -setnumber ...
        LY 1 <fc-oogmsh>/geodir/2d/rectanglepart.geo -o <fc-oogmsh>/meshes/rectanglepart-Lx1.000-Ly1.000-Nx3-Ny2-N100.msh
 Be patient...
```

Octave commands with output

```
pmfile=gmsh.buildpartrectangle(1,1,3,2,100,'verbose',4, ...
     'force',true,'meshfile','./toto.msh');;
```

```
[fc-oogmsh] Input file : <fc-oogmsh>/geodir/2d/rectanglepart.geo
[fc-oogmsh] Starting building mesh ./toto.msh with gmsh 4.0.1
[fc-oogmsh] Using command : gmsh -2 -string "Mesh.MshFileVersion=2;" -setnumber N 100 -setnumber NX 3 -setnumber NY 2 -setnumber LX 1 -setnumber ...
      LY 1 <fc-oogmsh>/geodir/2d/rectanglepart.geo -o ./toto.msh
 Be patient...
[fc-oogmsh] gmsh output :
Info    : Running '/home/cuvelier/bin/gmsh -2 -string Mesh.MshFileVersion=2; -setnumber N 100 -setnumber NX 3 -setnumber NY 2 -setnumber LX 1 ...
      -setnumber LY 1 <fc-oogmsh>/geodir/2d/rectanglepart.geo -o ./toto.msh' [Gmsh 4.0.1, 1 node, max. 1 thread]
Info    : Started on Wed Sep 26 08:30:06 2018
Info    : Reading '<fc-oogmsh>/geodir/2d/rectanglepart.geo'...
Info    : Reading '<fc-oogmsh>/geodir/2d/partitions01_data.geo'...
Info    : Done reading '<fc-oogmsh>/geodir/2d/partitions01_data.geo'
Info    : Reading '<fc-oogmsh>/geodir/2d/partitions_shape.geo'...
Info    : Done reading '<fc-oogmsh>/geodir/2d/partitions_shape.geo'
Info    : Done reading '<fc-oogmsh>/geodir/2d/rectanglepart.geo'
Info    : Meshing 1D...
Info    : Meshing curve 1 (Line)
Info    : Meshing curve 2 (Line)
Info    : Meshing curve 3 (Line)
Info    : Meshing curve 4 (Line)
Info    : Meshing curve 5 (Line)
Info    : Meshing curve 6 (Line)
Info    : Meshing curve 7 (Line)
Info    : Meshing curve 8 (Line)
Info    : Meshing curve 9 (Line)
Info    : Meshing curve 10 (Line)
Info    : Meshing curve 11 (Line)
Info    : Meshing curve 12 (Line)
Info    : Meshing curve 13 (Line)
Info    : Meshing curve 14 (Line)
Info    : Meshing curve 15 (Line)
Info    : Meshing curve 16 (Line)
Info    : Meshing curve 17 (Line)
Info    : Done meshing 1D (0.005222 s)
Info    : Meshing 2D...
Info    : Meshing surface 19 (Plane, Delaunay)
Info    : Meshing surface 21 (Plane, Delaunay)
Info    : Meshing surface 23 (Plane, Delaunay)
Info    : Meshing surface 25 (Plane, Delaunay)
Info    : Meshing surface 27 (Plane, Delaunay)
Info    : Meshing surface 29 (Plane, Delaunay)
Info    : Done meshing 2D (0.330483 s)
Info    : 13685 vertices 27682 elements
Info    : Writing './toto.msh'...
Info    : Done writing './toto.msh'
Info    : Stopped on Wed Sep 26 08:30:06 2018
```

4 ooGmsh class

The ooGmsh class can be used to read **gmsh** mesh files with the MSH ASCII file format described for example in [1], section 9.1.

In a .msh file the kind of mesh elements are identified by their *elm-type* integer values :

elm-type	**description**
1	2-node line
2	3-node triangle
3	4-node quadrangle
4	4-node tetrahedron
5	8-node hexahedron
6	6-node prism
7	5-node pyramid
8	3-node second order line (2 nodes associated with the vertices and 1 with the edge)
9	6-node second order triangle (3 nodes associated with the vertices and 3 with the edges)
10	9-node second order quadrangle (4 nodes associated with the vertices, 4 with the edges and 1 with the face)
11	10-node second order tetrahedron (4 nodes associated with the vertices and 6 with the edges)

12	27-node second order hexahedron (8 nodes associated with the vertices, 12 with the edges, 6 with the faces and 1 with the volume)
13	18-node second order prism (6 nodes associated with the vertices, 9 with the edges and 3 with the quadrangular faces)
14	14-node second order pyramid (5 nodes associated with the vertices, 8 with the edges and 1 with the quadrangular face)
15	1-node point
16	8-node second order quadrangle (4 nodes associated with the vertices and 4 with the edges)
17	20-node second order hexahedron (8 nodes associated with the vertices and 12 with the edges)
18	15-node second order prism (6 nodes associated with the vertices and 9 with the edges)
19	13-node second order pyramid (5 nodes associated with the vertices and 8 with the edges)
20	9-node third order incomplete triangle (3 nodes associated with the vertices, 6 with the edges)
21	10-node third order triangle (3 nodes associated with the vertices, 6 with the edges, 1 with the face)
22	12-node fourth order incomplete triangle (3 nodes associated with the vertices, 9 with the edges)
23	15-node fourth order triangle (3 nodes associated with the vertices, 9 with the edges, 3 with the face)
24	15-node fifth order incomplete triangle (3 nodes associated with the vertices, 12 with the edges)
25	21-node fifth order complete triangle (3 nodes associated with the vertices, 12 with the edges, 6 with the face)
26	4-node third order edge (2 nodes associated with the vertices, 2 internal to the edge)
27	5-node fourth order edge (2 nodes associated with the vertices, 3 internal to the edge)
28	6-node fifth order edge (2 nodes associated with the vertices, 4 internal to the edge)
29	20-node third order tetrahedron (4 nodes associated with the vertices, 12 with the edges, 4 with the faces)
30	35-node fourth order tetrahedron (4 nodes associated with the vertices, 18 with the edges, 12 with the faces, 1 in the volume)
31	56-node fifth order tetrahedron (4 nodes associated with the vertices, 24 with the edges, 24 with the faces, 4 in the volume)
92	64-node third order hexahedron (8 nodes associated with the vertices, 24 with the edges, 24 with the faces, 8 in the volume)
93	125-node fourth order hexahedron (8 nodes associated with the vertices, 36 with the edges, 54 with the faces, 27 in the volume)

When reading a .msh file generated by `gmsh`, we split the mesh elements by *elm-type* and generate an array of ELMT structure. The dimension of this array is the number of differents *elm-type* founds on the .msh file. The `Elmt` structure is given by

Fields of `Elmt` structure

type	:	integer refers to the type of the element : 1 for 2-node line, 2 for 3-node triangle, ... See the *elm-type* description of [1], section 9.1.
geo	:	string contains the kind of geometry: 'line', 'triangle', 'tetrahedron', ...
d	:	integer space dimension or d-simplex.
order	:	integer order of the element
$\mathrm{n_{me}}$	:	integer number of mesh elements
me	:	array of $d+1$-by-$\mathrm{n_{me}}$ integers connectivity array
phys_lab	:	array of $\mathrm{n_{me}}$-by-... integers physical labels of the elements
geo_lab	:	array of $\mathrm{n_{me}}$-by-... integers geometrical labels of the elements
nb_parts	:	array of $\mathrm{n_{me}}$-by-1 integers number of mesh partitions to which the element belongs
part_lab	:	array of $\mathrm{n_{me}}$-by-max(nb_parts) integers part_lab$(i, 1 :$ nb_parts$(i))$ contains all the partitions index to which the i-th element belongs.

The OOGMSH class was created to store a maximum of(all the) information(s) contained in the .msh file. The properties of this class are:

Properties of ooGmsh class

dim	:	integer space dimension
n_q	:	integer number of vertices/nodes
q	:	dim-by-n_q array of reals array of vertex coordinates
types	:	array of integers List of the element types found in the mesh file.
orders	:	array of integers List of the orders of the element types found in the mesh file.
sElts	:	array of `Elmt` structure One `Elmt` structure by element type, such that sElts(i) contains all the elements of type types(i) and order orders(i).

The ooGmsh class have only one constructor :

```
Gh=ooGmsh(meshfile)
```

where meshfile is the name of ... a mesh file

4.1 Sample 1

The 2d .geo file *condenser.geo* is used to create a .msh file : `condenser-25.msh`. This `.msh` file contains only 1 (2-node line) and 2 (3-node triangle) *elm-type*.

Octave commands with output

```
meshfile=gmsh.buildmesh('condenser',25,'verbose',0);
Gh = ooGmsh(meshfile)
```

```
Gh =

  ooGmsh with properties:
               dim: 2 double
                nq: 55651 double
            orders: 1 double
    partitionnedfile: 0 logical
                 q: (2x55651 double)
             sElts: (3x1 struct)
          toGlobal: (1x55651 double)
             types: [ 1 2 15 ] (1x3 double)
```

4.2 Sample 2

The 3d .geo file *cylinderkey.geo* is used to create a .msh file : `cylinderkey-10.msh`. This `.msh` file contains 1 (2-node line), 2 (3-node triangle) and 4 (4-node tetrahedron) *elm-type*.

Octave commands with output

```
meshfile=gmsh.buildmesh3d('cylinderkey',10,'verbose',0,'force',true);
Gh = ooGmsh(meshfile)
```

```
Gh =

  ooGmsh with properties:
              dim: 3 double
               nq: 5132 double
           orders: 1 double
   partitionnedfile: 0 logical
                q: (3x5132 double)
            sElts: (3x1 struct)
         toGlobal: (1x5132 double)
            types: [ 1 2 4 ] (1x3 double)
```

4.3 Sample 3

The 3d .geo file *ball8.geo* is used to create a 3d surface .msh file : `ball8-50.msh`. This `.msh` file contains 1 (2-node line), 2 (3-node triangle) and 15 (1-node point) *elm-type*.

Octave commands with output

```
meshfile=gmsh.buildmesh3ds('ball8',50,'verbose',0,'force',true);
Gh = ooGmsh(meshfile)
```

```
Gh =

  ooGmsh with properties:
              dim: 3 double
               nq: 37137 double
           orders: 1 double
   partitionnedfile: 0 logical
                q: (3x37137 double)
            sElts: (3x1 struct)
         toGlobal: (1x37137 double)
            types: [ 1 2 15 ] (1x3 double)
```

4 References

[1] Gmsh 2.15.0. `http://gmsh.info`, 2016.

[2] C. Geuzaine and J.-F. Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.