A Gmsh tutorial

A. Avdis and S.L.Mouradian
AMCG

Summary

This document is a tutorial on the Gmsh mesh generator. It is aimed towards complete beginners; only some basic knowledge of the Linux terminal and a text editor is assumed. We first define what a mesh is and then introduce the reader to the basics of the Gmsh graphical user interface. A basic, two-dimensional, geometry is then constructed within Gmsh and a mesh is constructed. A more complicated three-dimensional annulus is also constructed and meshed, demonstrating some more advanced features of Gmsh.
Having mastered the basic usage of the graphical user interface, users are introduced to generating simple meshes on the sphere. Finally, other tutorials and methds that show how to produce meshes in realistic domains are briefly introduced in the last section.

Contents

This document was compiled on November 3, 2012

url to the latest version:

http://amcg.ese.ic.ac.uk/files/gmsh_tutorial.pdf

1 Introduction

1.1 What is a mesh?

A mesh can be qualitatively thought of as the tessellation of a domain Ω into a set of non-overlapping subdomains ω_i:

$$\Omega = \cup \{\omega_i \,|i = 1, 2, \ldots ele\}$$
$$0 = \cap \{\omega_i \,|i = 1, 2, \ldots ele\} \tag{1}$$

where ele is the number of elements in the tessellation. Figure 1 shows meshes on one-dimensional and two-dimensional domains. The extension to three-dimensional domains is clear. Whatever the dimensionality of the domain, it is evident from figure 1 that the mesh can be constructed by first distributing a set of nodes throughout the domain, and then connect the nodes, so as to obtain a set of non–overlapping elements. Therefore, in order to define a mesh, one needs to define node locations, as well as element topology consistent to equations (1).

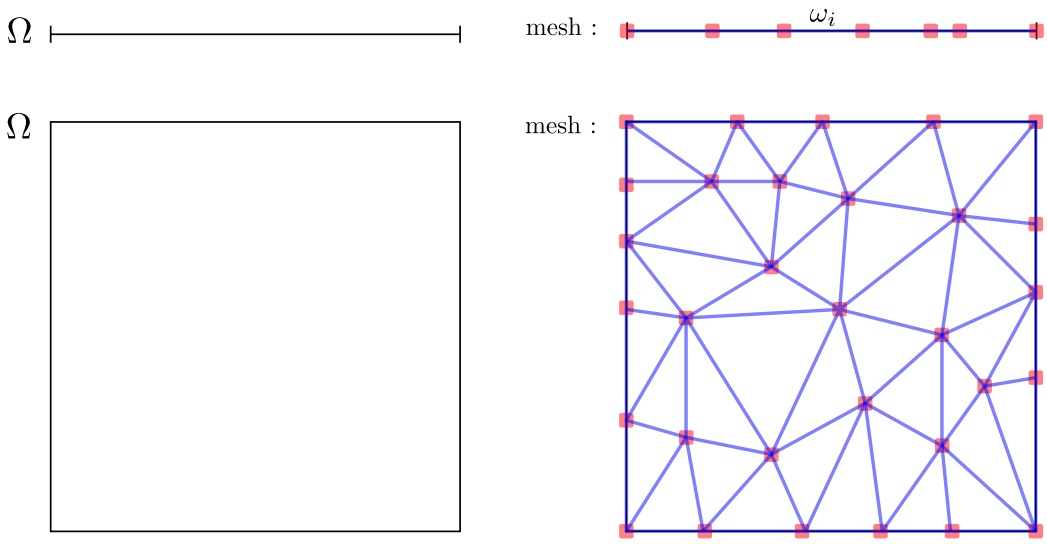


Figure 1: *Examples of meshes in one-dimensional and two-dimensional domains.*

1.2 What is Gmsh?

It is the task of a *mesh generator* to create node locations and element topology so as to create high quality meshes. Gmsh is a *"3D finite element grid generator with a build-in CAD engine and post-processor. Its design goal is to provide a fast, light and user-friendly meshing tool with parametric*

input and advanced visualisation capabilities."[1]. Furthermore, Gmsh can be used as a 1–, 2– and 3–dimensional mesh generator for use with Fluidity.

Gmsh is developed by Geuzaine and Remacle (2009) and distributed under the terms of the GNU General Public License. Executables for Linux, Windows and Mac OS can be downloaded from `http://www.geuz.org/gmsh/` as well as the source code. In Linux distributions where the Advanced Package Tool (APT) is available (such as Ubuntu), typing `sudo apt-get install gmsh` in a terminal will probably install Gmsh on your system. The rest of this tutorial assumes that Gmsh is run on Linux.

Gmsh's architecture is centred around four modules:

1. Geometry
2. Mesh
3. Solver
4. Post-processing

In this tutorial we will only use the Geometry and Mesh modules. As suggested by their names the geometry module is used to define geometrical objects, such as points, lines, surfaces and volumes, while the mesh module is used to create meshes (nodes and element topology).

1.2.1 Starting Gmsh

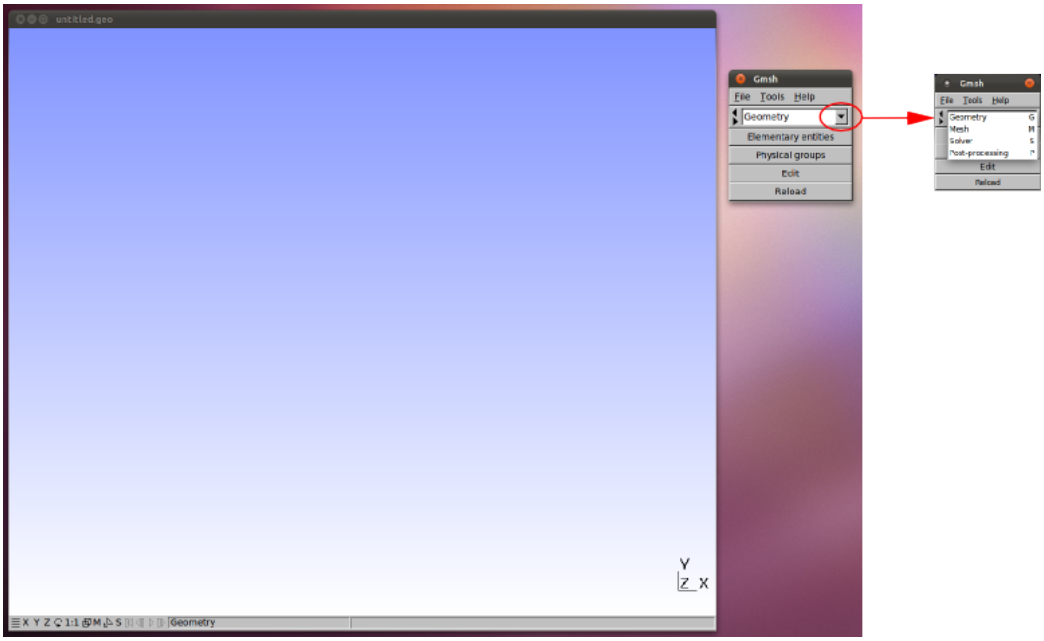


Figure 2: *Left panel: The Gmsh windows after startup. The larger window is termed the Graphic window and the smaller window is termed the Menu window.*

[1] from `http://www.geuz.org/gmsh/`

Gmsh can be started by issuing the following command at the Terminal:

```
gmsh mymesh.geo
```

where `mymesh.geo` is the name of the newly created file to store your geometry. If a file is not named, an `untitled.geo` file will be created. For this tutorial, it is best practice to name your files and avoid using an "untitled" document. As the Gmsh user interacts with the GUI, her/his actions are stored in the `.geo` file, in the form of script commands. One can modify the script with any text editor. Once Gmsh is opened, you will be presented with the Gmsh *menu* and *graphic* windows, as illustrated in figure 2.

1.2.2 Basic interaction with the Graphical User Interface

Once Gmsh is opened, you will be presented with the Gmsh *menu* and *graphic* windows, as illustrated in figure 2. The *menu window* allows the user to access features of the Gmsh graphical user interface, while the *graphic window* displays the defined points, lines and any other geometrical objects. The modular architecture of Gmsh is reflected in the menu window; clicking on the down-pointing arrow next to "Geometry" produces a drop-down menu (as shown in figure 2), allowing the user to select any of the four modules listed in section 1.2. Familiarise yourself with the Gmsh menu window, by

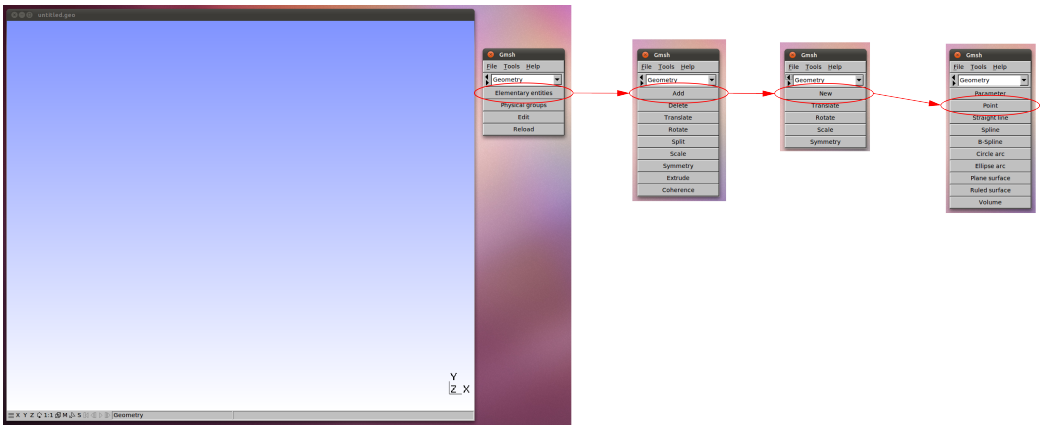


Figure 3: *Basic interaction with the GMSH GUI*

navigating to a mode that allows you to add a point. Try the following for yourself to get comfortable navigating the menu and understand the notation provided above, which will be used through this tutorial to point you to specific features of Gmsh.

1. Click on `Geometry (G) > Elementary entities > Add > New > Point`, as indicated in figure 3. After clicking on "Point" in the menu window the "Contextual Geometry Definitions" window will appear and a message appears at the top of the Graphic window, as shown in figure 4.

2. Close the Contextual Geometry Definitions window and press "q" to abort the command.

3. You may use the keyboard buttons G, M, S and P to get to the top-level of the Geometry, Mesh, Solvers and Post-processing modules respectively. Press G to return to the top-level of the Geometry module. Try switching to the Mesh module and then back to the Geometry module.

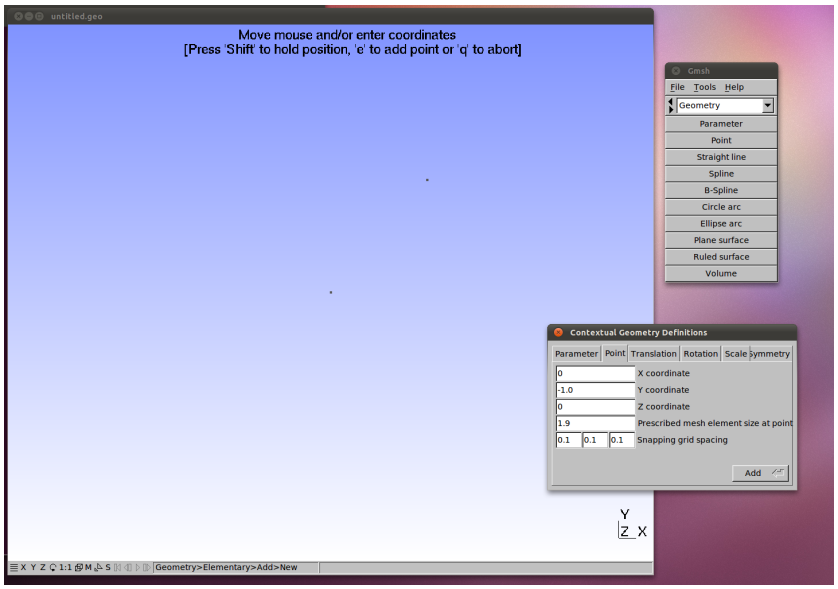


Figure 4: *Creating a point.*

We will now open an existing geometry and learn to navigate by panning, zooming and rotating. To obtain the geometry, please download the file at:

`http://amcg.ese.ic.ac.uk/download/annulus.geo`

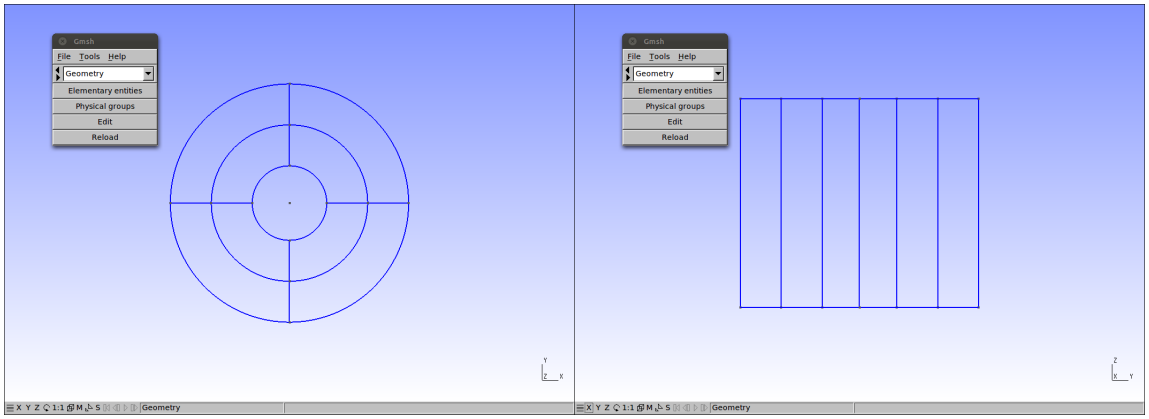
and open it by issuing the following at the command line:

```
gmsh annulus.geo
```

Once open, you should see the geometry illustrated in figure 5a. On the bottom-left of the graphic window, you will see various tools available to manage the view. The X, Y and Z buttons align the corresponding axis perpendicular to the screen. Try selecting the X view, as in figure 5b. Both these views are not particularly useful in understanding the annulus domain. Try to rotate the geometry by left-clicking and holding on the screen, then moving the mouse pointer. Try to get the geometry as illustrated in figure 5c. You can pan the geometry by right-clicking and moving the mouse. To zoom, simply use the scroll. Alternatively, you may hold the middle-button and move the mouse. Make sure you familiarise yourself with manipulating the orientation of your domain, as this will be useful later on.

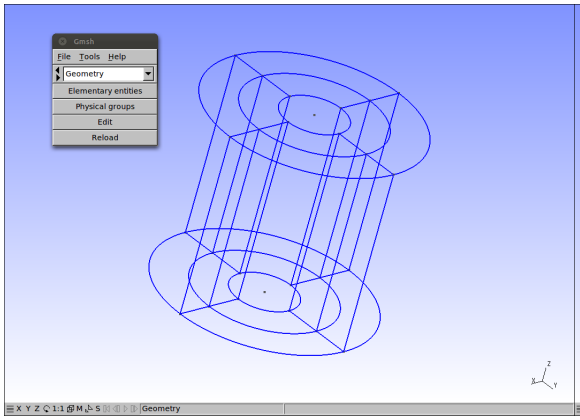
As no meshing tutorial can be complete without a mesh, we end by meshing the geometry. To generate a mesh of this domain, go to:

```
Mesh (M) > 3D
```

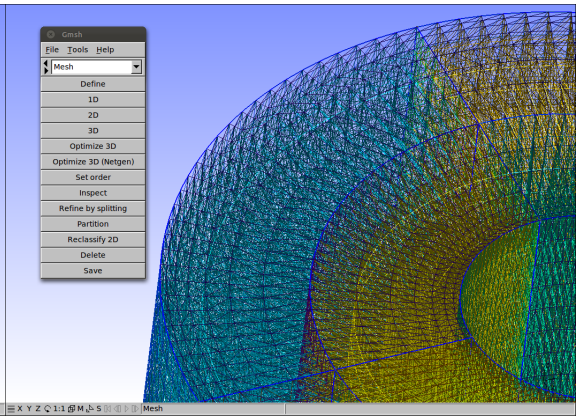


(a) Default view

(b) x–axis perpendicular to screen



(c) Rotated view



(d) Meshed and zoomed

Figure 5: *Panning, zooming, rotating and manipulating the Gmsh graphic window.*

This can be done by pressing your keyboard's M key, and then selecting $3D$ to mesh the three–dimensional domain. Figure 5d shows a view of the meshed annulus.

2 A two dimensional example

In this example we aim to create a mesh in a rectangle, as shown in figure 6. The following steps will

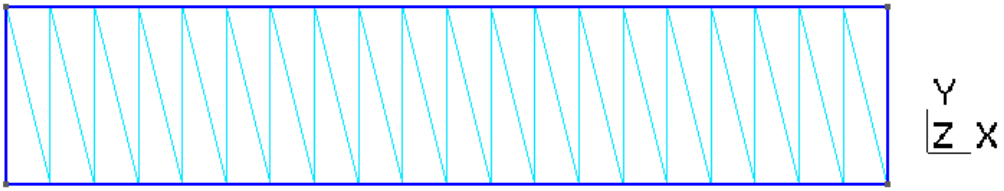


Figure 6: *Mesh on the rectangle*

be taken:

1. Set up the geometry. The graphical user interface of Gmsh is used to carry out the following steps:
   (a) Create the two points on the left-hand side of the rectangle.
   (b) Connect the two points with a line
   (c) Extrude the line to form the rectangle.
2. Define physical groups. Defining physical groups results into numerical ID's being assigned to each group. Fluidity uses these ID's to apply boundary conditions.
3. Customise the geometry. A subset of the Gmsh scripting language is introduced and modifications will be made to the geometry script.
4. Produce a mesh.

The following sections go through the above steps, in great detail.

2.1 Setting up the geometry

Run gmsh on the command line, storing the geometry in `channel.geo`:

```
gmsh channel.geo
```

Create the points at the left-hand-side of the rectangle:

1. Go to the top-most level of the Geometry module in the menu window.
2. On the menu window click on `Elementary entities > Add > New > Point` , see figure 3. The "Contextual Geometry Definitions" window will appear at the point tab, as shown in figure 4. Note that moving the cursor over the graphic window will cause the coordinates shown in the Contextual Geometry Definitions window to change. While the user is allowed to specify points using this interactive & graphical interface, we will directly enter the point coordinates

on the Contextual Geometry Definitions window. Doing so allows for more accurate control of the point coordinates.

3. Add a point at $(x, y, z) = (0, -1, 0)$, with a characteristic mesh element size of 1.9: Modify the parameters in the Contextual Geometry Definitions window, to match those shown in figure 4 and click on "Apply" (on the same window). Be careful not to move the cursor over the graphic window, as it will change your coordinates.

4. Add another point at $(0, 1, 0)$

5. Close the Contextual Geometry Definitions window.

Create a line representing the left-hand edge of the rectangle:

1. Click on `Geometry (G) > Elementary entities > Add > New > Straight line`

2. Choose the starting point of the line segment on the graphic window. Note that the starting point should now be coloured red. If not, try selecting it again.

3. Choose the ending point of the line segment. If the point selection is successful, the straight line segment should appear in the graphic window, and you should have something similar to the left-most panel in figure 7.

4. Press "q" on your keyboard.

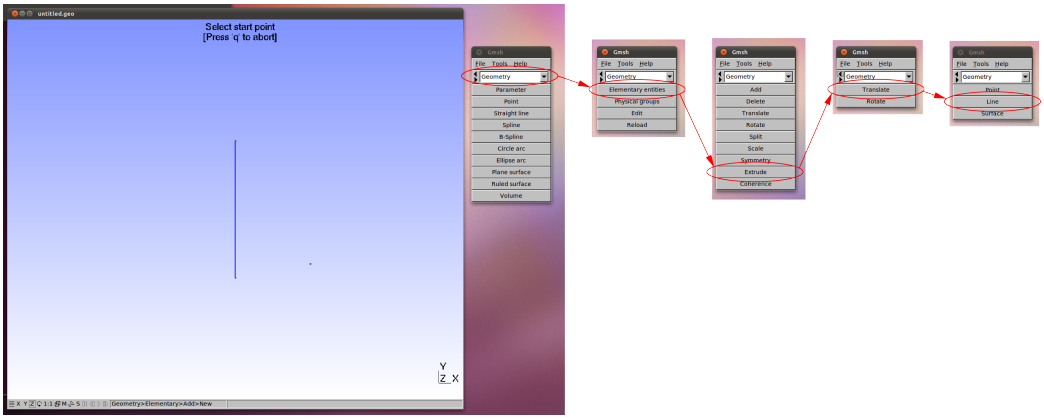


Figure 7: *Directing Gmsh to extrusion.*

Note that one could simply create all four points at the rectangle vertices and then simply draw the lines to from the rectangle sides. However, extrusions are used below as it demonstrates more features of Gmsh and it facilitates the creation of a structured mesh:

1. Go to the top-most level of the Geometry module in the menu window.

2. Click on `Elementary entities > Extrude > Translate > Line`, as indicated in figure 7. The Contextual Geometry Definitions window will appear, set to the Translation tab, as shown in the left panel of figure 8

3. Modify the parameters in the Contextual Geometry Definitions window to specify an extrusion along the x-direction of 10 length-units, as shown in the left panel of figure 8.

4. Click on the line we previously drew on the graphical window, it should be highlighted in red, as shown in the left panel of figure 8.

5. Press the key "e" on your keyboard, a rectangle should appear in the graphical window, as shown in the right panel of figure 8.

6. Close the Contextual Geometry Definitions window.

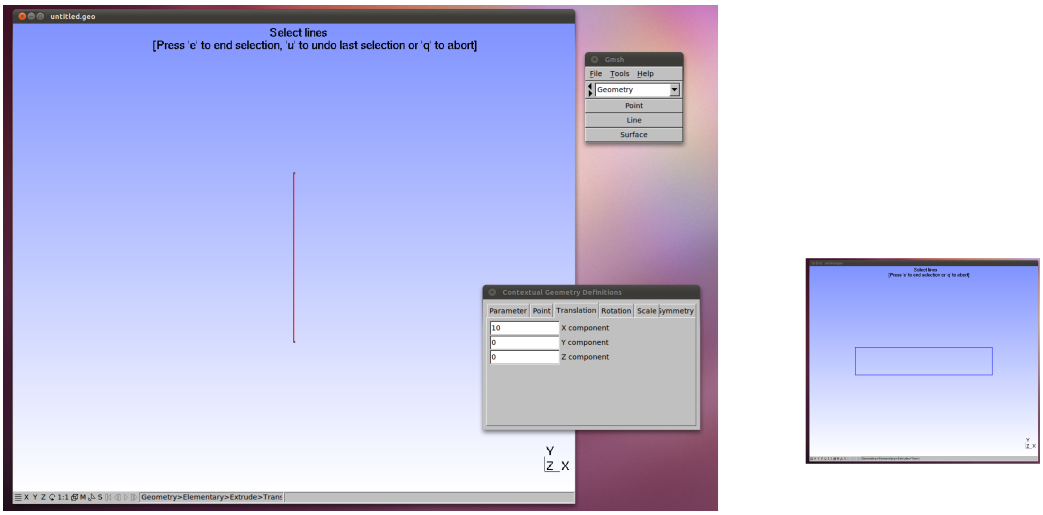


Figure 8: *Extruding a line to form a rectangle.*

2.2 Physical groups: boundaries and regions

In order to apply boundary conditions it is necessary to specify "Physical groups" to which the boundaries belong. Since Gmsh will only export to a file those elements which are associated with a physical entity, it is also necessary to identify the interior of the domain with a physical group. Declare a physical surface over the interior of the rectangle as follows:

1. Go to the top-most level of the Geometry module in the menu window.

2. Click on `Geometry > Physical group > Add > Surface`.

3. Gmsh will highlight any surfaces with grey broken lines, see figure 9 for example.

4. Select the plain-surface created in section 2.1, by clicking on the broken grey line.

5. Press "e" on your keyboard to end the selection, and then "q".

Then declare physical lines from the edges of the rectangle as follows:

1. On the menu window of Gmsh click on “Line”, pick the top side of the rectangle and press “e” on your keyboard to end the selection.
2. Pick the top side and the bottom side of the rectangle and press “e” on your keyboard to end the selection.
3. On the menu window of Gmsh click on “Line”, pick the left-hand side of the rectangle and
4. Repeat the above step, selecting the right-hand-side of the rectangle.
5. press “q” to exit the selection mode.

2.3 Final customisation of the geometry

So far, we have been using the graphical user interface to create a simple geometry. we will now use a less-graphical oriented approach to adjust some important details. At the beginning of the section Gmsh was invoked using file `channel.geo`. The file `channel.geo` includes a set of declarations that allow Gmsh to reconstruct the geometry, effectively allowing the user to “save” the geometry. Since the `channel.geo` file is an ASCII file, any text editor can be used to edit it, however, we will here use the text editor invoked by Gmsh itself:

1. Go to the top-most level of the geometry module in the Gmsh menu window.
2. Click on `Edit`, a text editor instance (usually emacs) will open the file `channel.geo`, as shown in figure 9.

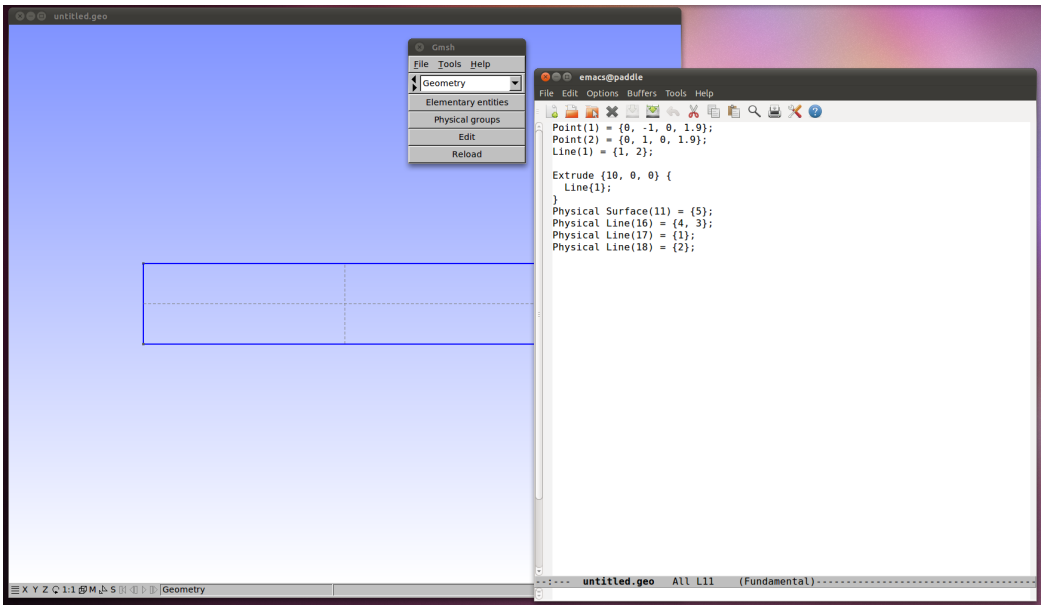


Figure 9: *The Gmsh windows with the text editor invoked on* `channel.geo`

The file contents should be identical to the left hand-side of the listing below. The numbers present on the left hand-side of the listings are added here for ease of reference. The `.geo` files are script files

containing comments and expressions. Expressions can be further classified into string-expressions, floating-point expressions Loops & Conditionals as well as other categories, listing the categories is beyond the scope of this tutorial (however, the interested reader can find the full documentation at http://geuz.org/gmsh/doc/texinfo/gmsh.html#General-tools). The expressions of interest to this tutorial are commands that define geometrical objects. So, before the reader proceeds with editing the file contents, we briefly describe the commands and the syntax of comments.

- Lines preceded by `//` are comment lines and are ignored by Gmsh as it reads the file. Also any part of a script file between `/*` and `*/` is treated as a comment.
- Commands must terminate with `;`, otherwise Gmsh assumes that the command is continued in the line below (see lines 5 and 6).[2]
- Commands such as those in line 1, clearly create a point, and:
  - Assign the point-ID in the round brackets on the left-hand-side of the expression to the point with coordinates specified by the first three numbers in the braces.
  - Assign the fourth number in the braces as characteristic mesh size.
- Commands such as those in line 3, clearly create a line segment, and:
  - Assign the line-ID in the round brackets on the left-hand-side of the expression to the line with endpoint specified by the point-IDs in the braces.
- The extrusion command in line 5 created the three remaining lines of the rectangle, as well as the plane surface

Edit the file, so as to match the right hand side listing.

`channel.geo` file as produced by GUI.

```
1  Point(1) = {0, -1, 0, 1.9};
2  Point(2) = {0, 1, 0, 1.9};
3  Line(1) = {1, 2};
4
5  Extrude {10, 0, 0} {
6    Line{1};
7  }
8  Physical Surface(11) = {5};
9  Physical Line(16) = {4, 3};
10 Physical Line(17) = {1};
11 Physical Line(18) = {2};
```

`channel.geo` file after fine-tuning.

```
1  Point (1) = {0, -1, 0, 1.9};
2  Point (2) = {0, 1, 0, 1.9};
3  Line (1) = {1, 2};
4
5  Extrude {10, 0, 0} {
6    Line{1};Layers{20};
7  }
8
9  // Volume number for whole domain.
10 Physical Surface (1) = {5};
11 // Top and bottom of the box.
12 Physical Line(333) = {3,4};
13 // Left wall
14 Physical Line(666) = {1};
15 // Right wall
16 Physical Line(444) = {2};
```

In particular the following changes should be made:

[2] Also note that at the end of line 7 a semi-colon should be present. Currently, its absence does not seem to affect Gmsh.

1. Append `Layers{20};` to line 6. This instructs Gmsh to organise the mesh in 20 layers, normal to the extrusion direction.
2. In line 9 change `Physical Line(16)` to `Physical Line(333)`
3. In line 10 change `Physical Line(17)` to `Physical Line(666)`
4. In line 11 change `Physical Line(18)` to `Physical Line(444)`
5. Insert comment-lines to match the left-hand-side listing above.

Save the file and close the editor window. Click on `Reload` in the Gmsh menu window, Gmsh needs to read the updated file. You are now ready to generate a mesh.

2.4 Producing a mesh

For the present example, generating a mesh once the geometry is properly defined is very simple:

1. On the Gmsh menu window go to the mesh module (or press the “m” key).
2. Click on “2D”. A mesh should appear inside the rectangle, as shown in figure 10.
3. Save the mesh: Click on “File”, then on “Save mesh” on the drop-down menu. The mesh is now saved in a file called `channel.msh`

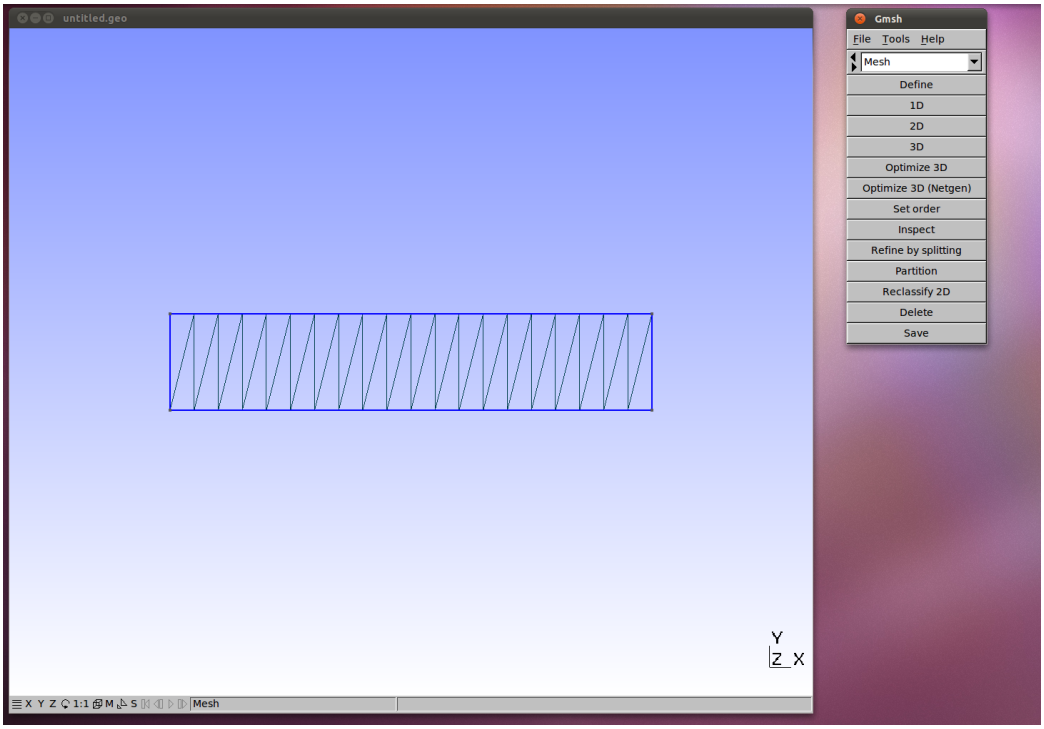


Figure 10: *The Gmsh windows showing the mesh.*

3 A three-dimensional, structured mesh example

The reader should be familiar with the geometry they will be creating in this section; the annulus the reader opened and meshed in section 1.2.2 is the geometry we will now create. The mesh however, will be of higher resolution. The following steps will be undertaken, also outlined in figure 11:

1. Create the geometry using the Gmsh GUI:
   (a) A rectangle will be formed using a procedure similar to section 2. This rectangle represents a radial plane through the annulus, for example the plane outlined in red broken lines in the left panel of figure 11.
   (b) The plane will then be extruded-rotated in order to form a quadrant of the annulus, as shown in the right panel of figure 11.
   (c) The above step is then repeated, as indicated by the arrows in the top of the annulus on the left panel of figure 11, until the complete annulus is formed
2. Define physical groups.
3. Customise the geometry by editing the geometry script file.
4. Produce a mesh.

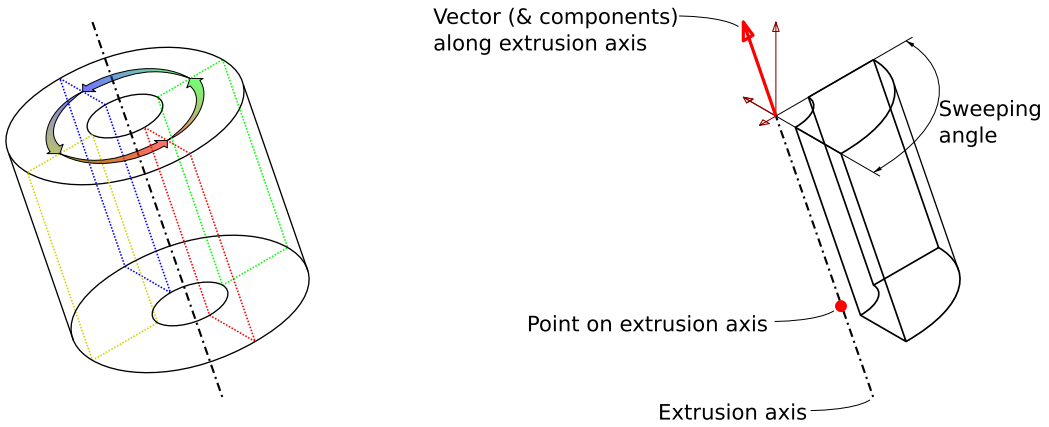


Figure 11: *Schematic of forming an annulus with extrusions*

3.1 Creating the geometry: Forming an annulus with extrusions

Start Gmsh, at the command line using file `annulus.geo` to store the script. Create the three points at the coordinates shown in table 1 below. The points should lie on a line along the x-direction and form a radial line across the annulus. Create two lines: the first line connecting point 1 to point 2 and the second line connecting the point 2 to point 3. Then, extrude the lines to create surfaces (extrude-translate, see figures 7 and 8), translating by $(x, y, z) = (0.0, 0.0, 14.0)$. Note that both lines should be selected during the extrusion step. Finally, change the view-point, so that you can see the rectangles

just created, you should have something similar to the graphic window in figure 12, corresponding to the plane highlighted by the red broken line in the left panel of figure 11

Table 1: *Coordinates and mesh size at the starting points when making an annulus.*

point ID	x-coord.	y-coord.	z-coord.	Mesh element size
1	2.5	0.0	0.0	0.1
2	5.25	0.0	0.0	1.0
3	8.0	0.0	0.0	0.1

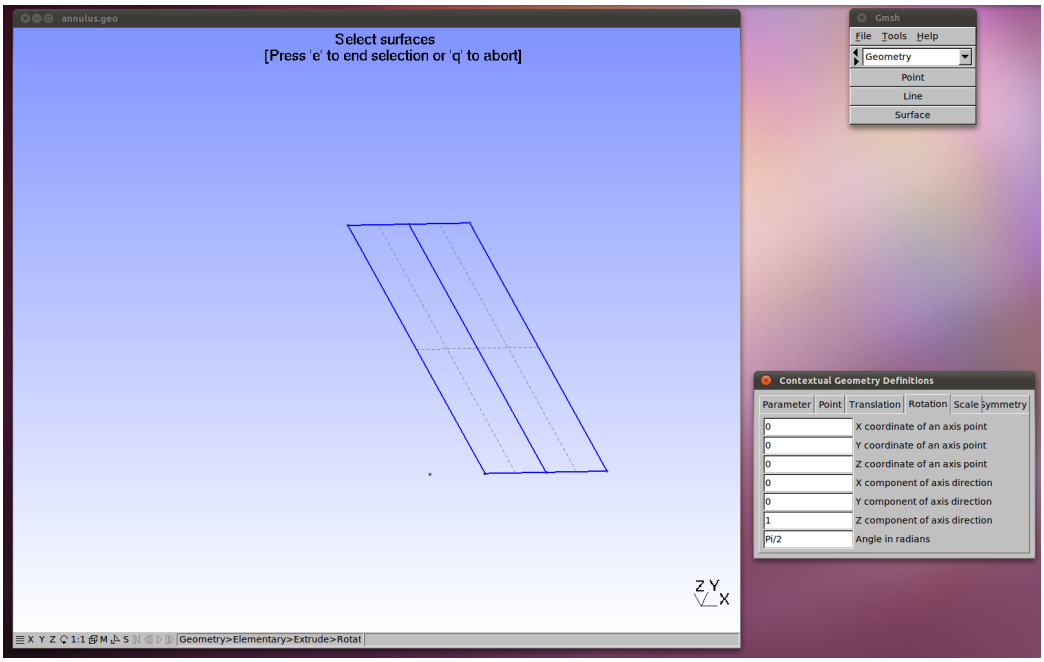


Figure 12: *The Graphic, Menu and Contextual Geometry Definitions windows during an extrusion-via-revolution step.*

The next step is to extrude-rotate the rectangular surface in order to create the annulus.

1. Go to the top-most level of the Geometry module in the Gmsh menu window.

2. Click on `Elementary entities > Extrude > Rotate > Surface`. The Contextual Geometry Definitions window will appear, on the Rotation tab. Also the surfaces will now be highlighted on the graphic window, as shown in the graphic window of figure 12. The Contextual Geometry Definitions window is used to define the axis of revolution and the sweeping angle. The axis of revolution is defined by specifying any point on it and the components of a vector parallel to the axis. (*e.g.* the red point and red vector components in the right panel of figure 11). In addition, the sweeping angle must be specified in radians, in the anti-clockwise sense.

3. Change the parameters in the Contextual Definitions window to the following:

   - X coordinate of an axis point: 0

- Y coordinate of an axis point: 0
- Z coordinate of an axis point: 0
- X component of axis direction: 0
- Y component of axis direction: 0
- Z component of axis direction: 1
- Angle in radians: Pi/2

4. Pick both surfaces on the graphic window and press “e”. A quarter of the annulus should have been formed in the graphic window, as shown in left panel of figure 13. Without changing the parameters in the Contextual Geometry Definitions window, pick the newly formed surfaces normal to the x-axis (surface demarcated in green in figure 11) and press the “e” key, to form half the annulus. Repeat the procedure to form the complete annulus, shown in the right panel of figure 13.

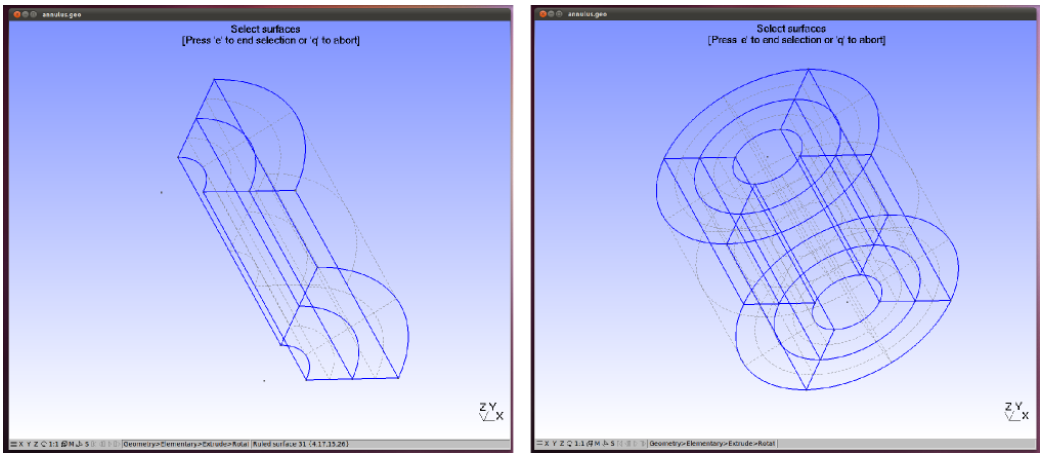


Figure 13: *The Gmsh window showing a quadrant (left panel) and the complete annulus (right panel).*

3.2 Physical groups

As in the previous section physical surfaces and volumes must be specified, start by specifying physical surfaces:

1. Go to the top-most level of the Geometry module in the Gmsh menu window.

2. Click on `Physical groups > Add > Surface`. Select the 8 surfaces comprising the top of the annulus and press “e”. Obviously you can pan, zoom and change your viewpoint to make selection easier. Press “u” to undo your last selection as hinted on the graphic window, if a wrong choice is made.

3. Repeat for the bottom of the annulus, the inner wall, and the outer wall (in that order).

4. Press “q” to abort the selection mode and end the command.

Then specify physical volumes:

1. Click on `Volume`, on the Gmsh menu window. Volumes created by extrusions will be indicated by yellow spherical markers, one per volume entity.
2. Pick all of the spherical markers. Once picked, a volume marker is highlighted in red.
3. Press “e” to end the selection.

3.3 Final customisation of the script and mesh production

At this point, a mesh can be produced, suitable for Fluidity simulations. The mesh will be unstructured and it is left as an exercise to the reader to mesh the geometry. Should you choose to mesh the geometry at this point, then once finished you must go to the top-most level of the Geometry module and click on `Reload` to re-join us for this tutorial. The script is now modified to make Gmsh produce a structured mesh:

1. Open the script in a text editor.
2. The script should be *similar* to the first listing below.
3. Edit the script to make it similar to the second listing below, and save it. You only have to append the `Layers{};` statements and insert the comments.
4. Go to the top-most level of the Geometry module in the Gmsh menu window. and click `Reload`.

`annulus.geo` file as produced by GUI.

```
Point(1) = {2.5, 0, 0, 0.1};
Point(2) = {5.25, 0, 0, 1.0};
Point(3) = {8, 0, 0, 0.1};
Line(1) = {1, 2};
Line(2) = {2, 3};
Extrude {0, 0, 14} {
  Line{1, 2};
}
Extrude {{0, 0, 1}, {0, 0, 0}, Pi/2} {
  Surface{6, 10};
}
Extrude {{0, 0, 1}, {0, 0, 0}, Pi/2} {
  Surface{32, 54};
}
Extrude {{0, 0, 1}, {0, 0, 0}, Pi/2} {
  Surface{98, 76};
}
Extrude {{0, 0, 1}, {0, 0, 0}, Pi/2} {
  Surface{120, 142};
}
Physical Surface(185) = {93, 49, 159, 115, 71, 27, 180, 137};
Physical Surface(186) = {107, 151, 41, 85, 63, 19, 172, 129};
```

```
Physical Surface(187) = {141, 184, 31, 75};
Physical Surface(188) = {89, 111, 155, 45};
Physical Volume(189) = {4, 3, 1, 2, 8, 7, 5, 6};
```

`annulus.geo` script after editing.

```
Point(1) = {2.5, 0, 0, 0.1};
Point(2) = {5.25, 0, 0, 1.0};
Point(3) = {8, 0, 0, 0.1};
Line(1) = {1, 2};
Line(2) = {2, 3};
Extrude {0, 0, 14} {
  Line{1, 2};Layers{25};
}
Extrude {{0, 0, 1}, {0, 0, 0}, Pi/2} {
  Surface{6, 10};Layers{30};
}
Extrude {{0, 0, 1}, {0, 0, 0}, Pi/2} {
  Surface{32, 54};Layers{30};
}
Extrude {{0, 0, 1}, {0, 0, 0}, Pi/2} {
  Surface{98, 76};Layers{30};
}
Extrude {{0, 0, 1}, {0, 0, 0}, Pi/2} {
  Surface{120, 142};Layers{30};
}
// Top
Physical Surface(185) = {93, 49, 159, 115, 71, 27, 180, 137};
// Bottom
Physical Surface(186) = {107, 151, 41, 85, 63, 19, 172, 129};
// Inner wall
Physical Surface(187) = {141, 184, 31, 75};
// Outer wall
Physical Surface(188) = {89, 111, 155, 45};
Physical Volume(189) = {4, 3, 1, 2, 8, 7, 5, 6};
```

To mesh the annulus, go to the top-most level of the Mesh module and click “3D” on the Gmsh menu window.Notice the gradation in cell size towards the wall, affected by a different characteristic mesh size in Point 2, see table 1 and the listings above.

4 Mesh generation on spherical manifolds

In this example, we generate a mesh of a section of the ocean on the surface of a sphere (the Earth). Within just a few steps, we will be able to produce a mesh similar to figure 14.

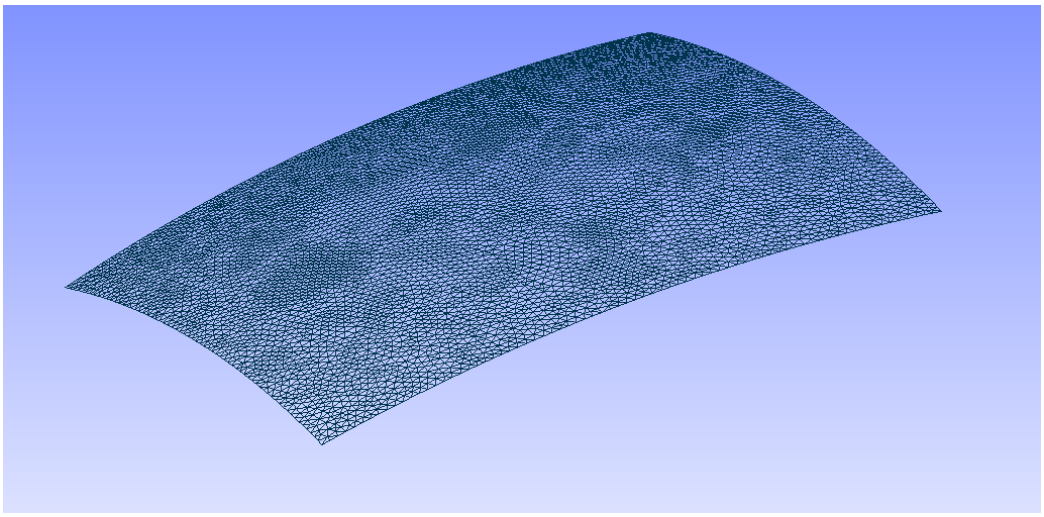


Figure 14: *A 2–D manifold in 3–D space.*

We will make primary use of the Gmsh scripting language and resort to the graphical user interface only to ensure that we are progressing correctly. We will be saving the script with the .geo extension, which Gmsh recognises as its own. To generate a file manifold.geo, we may issue:

```
gedit manifold.geo
```

which opens the gedit text editor and generates the manifold.geo file.

In this chapter you will learn how to:

1. Define points on the sphere (section 4.2).
2. Connect the points with lines (section 4.3).
3. Define physical surfaces and generate meshes on the sphere (section 4.4).

First, however, we will be introducing the process of stereographic projection (section 4.1). This is important because we need to define points on the projected plane before Gmsh automatically maps them back to the sphere.

4.1 Background: Stereographic projection

A stereographic projection, σ, maps all points (apart from one, the *North Pole*) of an n–dimensional unit sphere, $S^n \in R^{n+1}$, to an n–dimensional plane:

$$\sigma : S^n \rightarrow R^n \tag{2}$$

Figure 15 illustrates a two-dimensional sphere being projected onto a line.

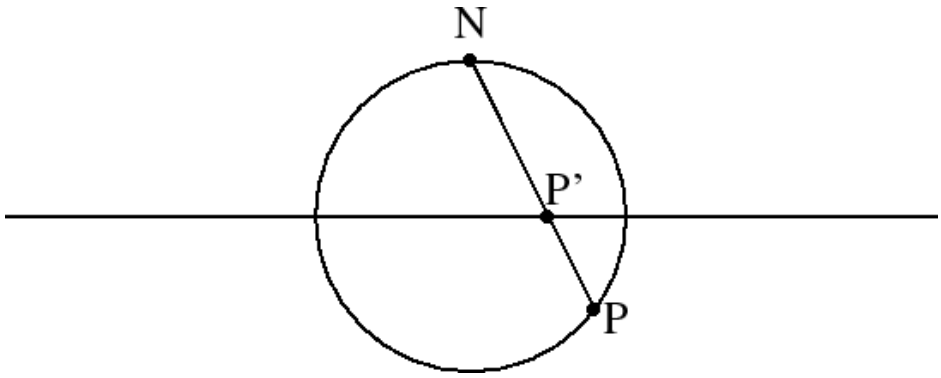


Figure 15: *2–D stereographic projection. Point N indicates the North Pole. Point P‘ is the projection of Point P.*

4.2 Defining Points

Gmsh has the capability of plotting points on the sphere. To the user‘s convenience, only the projections of these points need to be defined. Prior to these points, however, two special points are required:

1. The center of the sphere.
2. The *North Pole*.

To start our example, open a plain file and begin by defining the special points described:

```
Point(1) = {0.0,0.0,0.0};          //The center of the sphere
Point(2) = {0.0,0.0,6.37101e+06}; //The North Pole
```

We then need to inform Gmsh that any further points defined, will be on the sphere. So we issue the following statement:

```
PolarSphere(1) = {1,2}; //Sphere with center at Point(1)
                        //and North Pole at Point(2).
```

We now proceed by defining the remaining points, with coordinates on the 2–D plane. In this example, we will create a domain that spans from +10 to +30 degrees in longitude and +5 to +25 in latitude.

```
//Point 3: 10 lon 5 lat

//First we convert to radians
//and then to stereographic coordinates.
Point_lon_rad = (10)*(Pi/180);
Point_lat_rad = (5)*(Pi/180);
Point_3_stereographic_x = -Cos(Fabs(Point_lon_rad))*Cos(Fabs(Point_lat_rad))/
    ( 1 + Sin(Fabs(Point_lat_rad)) );
Point_3_stereographic_y = Cos(Fabs(Point_lat_rad))*Sin(Fabs(Point_lon_rad))/
    ( 1 + Sin(Fabs(Point_latx_rad)) );

//Now we define the point
Point(3) = {Point_stereographic_x, Point_stereographic_y,0};
```

We must repeat the process for the remaining three points:

- Point 4: 10 lon 25 lat

- Point 5: 30 lon 25 lat

- Point 6: 30 lon 5 lat

After we have defined the center, north pole and 4 edges of our domain; it is sensible to open up the file in Gmsh to have a look at our progress. We first save the file and then open it in Gmsh by issuing the following command at the terminal:

```
gmsh manifold.geo
```

The center and north pole should be visible, as well as the 4 points on the surface of the sphere, as in figure 16.

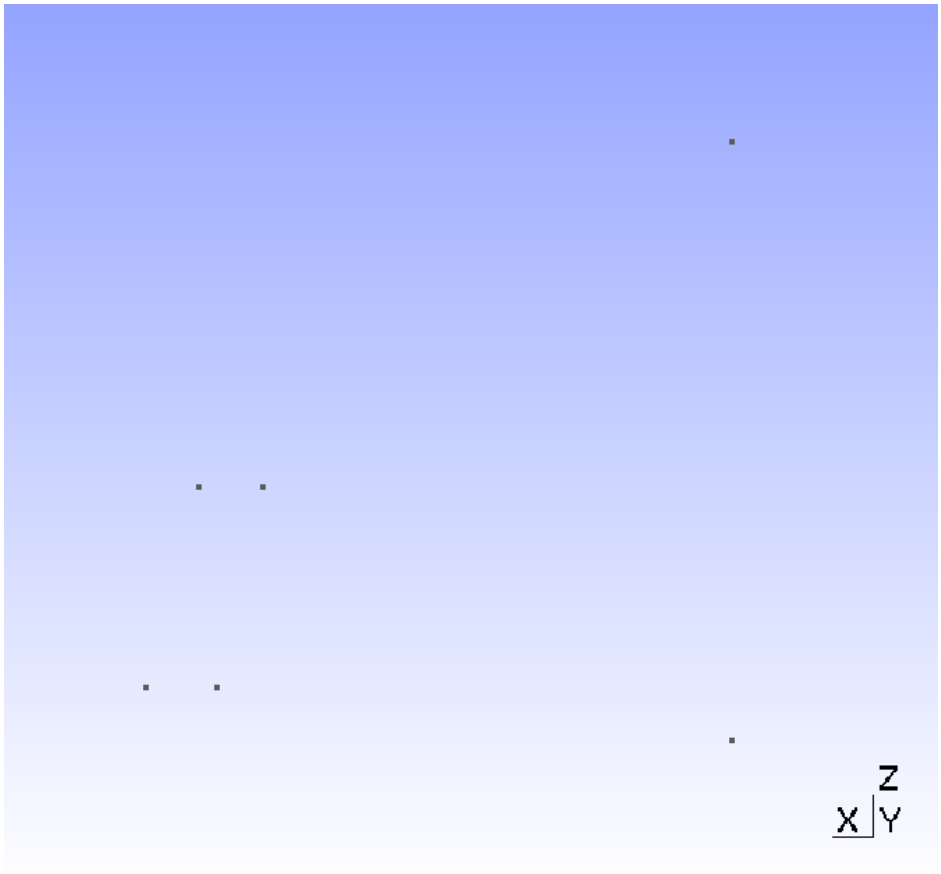


Figure 16: *The center of the sphere (below right), as well as the North pole (top right) and the four edges of the domain are visible in Gmsh. It is clear by looking at the coordinate system that the four points have been mapped from the 2–D plane.*

4.3 Defining zonal and meridional lines

In stereographic projection of a 2–D sphere such as the Earth, lines of constant latitude are mapped to straight lines in a plane. Gmsh is therefore able to accurately interpolate meridional lines, given just two points. Lines of constant longitude however, are projected to circles in the plane. The arcs between these points are also approximated by straight lines in Gmsh so a poor result is obtained if the distance between the points is large.

To correct this problem, we break any zonal paths into smaller segments for which the straight line assumption is valid. We will be using the following function:

```
Function DrawParallel
 alpha = parallelSectionStartingY/parallelSectionStartingX;
 parallelRadius = Sqrt(parallelSectionStartingX^2 + parallelSectionStartingY^2);
 deltaAlpha = (parallelSectionEndingY/parallelSectionEndingX -
    parallelSectionStartingY/parallelSectionStartingX)/pointsOnParallel;
 For t In {1:pointsOnParallel}
  alpha += deltaAlpha;
  newParallelPointX = (parallelSectionStartingX/Fabs(parallelSectionStartingX))
     *parallelRadius/(Sqrt(alpha^2 + 1));
  newParallelPointY = newParallelPointX*alpha;
  newPointOnParallel = newp;
  Point(newPointOnParallel) = {newParallelPointX, newParallelPointY, 0};
 EndFor
 BSpline(newParallelID) = {firstPointOnParallel, newPointOnParallel-
     (pointsOnParallel-1):newPointOnParallel, lastPointOnParallel};
Return
```

The function we have just defined requires the following parameters to be set prior to being called:

- pointsOnParallel : The number of points to be defined between the two we have already defined.
- parallelSectionStartingX : The first stereographic coordinate of the starting point. If we are drawing the southernmost zonal line, this will be the x-coordinate of the south-westernmost point (Point(3) in our case). The Y coordinate is also required in a respectively named parameter, as well as the X and Y coordinates of the ending point (south-easternmost point, Point(6)).
- firstPointonParallel : In addition to its coordinates, the function needs to know the point number of the first and last points of the zonal line.
- newParallelID : Finally, each line we draw needs a unique number.

We can now go ahead and plot the zonal lines by appending the following to our manifold.geo file:

```
//Draw south-most parallel of the Domain. Point 3 to Point 6

//Assign parameters to variables and then call function DrawParallel,
pointsOnParallel = 200;
parallelSectionStartingX = Point_3_stereographic_x;
parallelSectionStartingY = Point_3_stereographic_y;
firstPointOnParallel = 3;
```

```
parallelSectionEndingX = Point_6_stereographic_x;
parallelSectionEndingY = Point_6_stereographic_y;
lastPointOnParallel = 6;
newParallelID = 1;
Call DrawParallel;
```

The meridional lines are much simpler, and can be drawn by simply specifying to Gmsh that we require a B-spline between the two points:

```
//Draw western-most meridional
BSpline(3) = {3, 4};
//Draw eastern-most meridional
BSpline(4) = {5, 6};
```

Saving and opening the manifold.geo file in Gmsh should now look like what we see in figure 17.

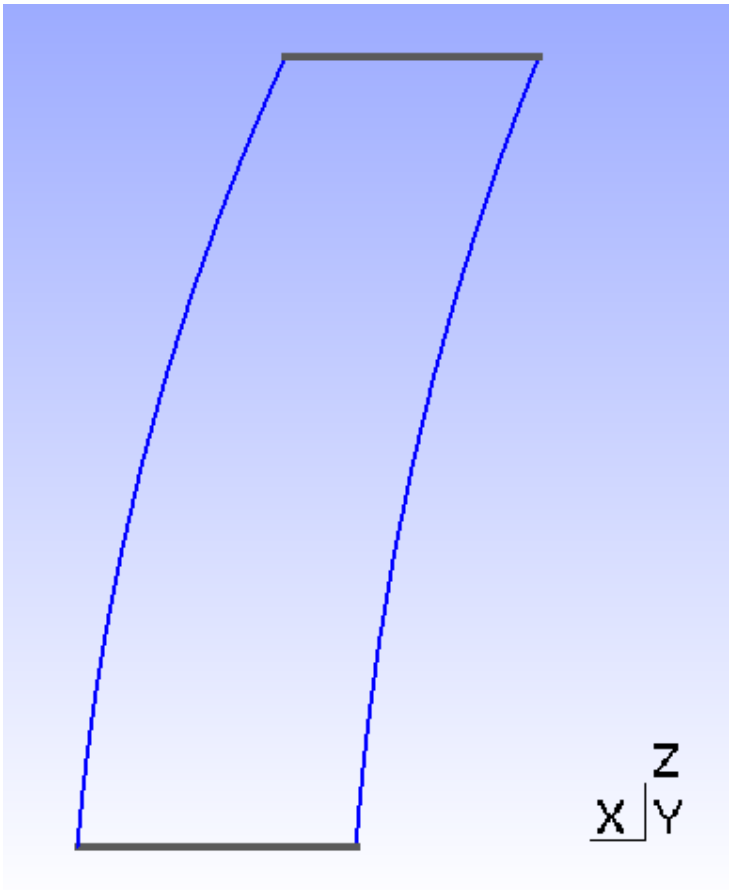


Figure 17: *The points of figure 16 have now been connected by lines. Note that zonal lines have had points defined along their path, as requested in the function we defined.*

4.4 Mesh generation

All that remains to generate the mesh is to specify the Physical surface and a Field to instruct Gmsh of the desired element length.

```
//Define new line from previously generated lines that generates a loop.
Line Loop(6) = {3,2,4,-1};
```

```
Plane Surface(7) = {6}; //Surface from Loop
Physical Surface(8) = {7};

Field[50] = MathEval; //Generate Field
Field[50].F = "30000";
Background Field = 50;
```

The variable `Field[50].F` should equal the desired element edge size. In our example, we set it equal to a constant `30000` indicating we want a constant element edge size of $30km$. This value could have been any other mathematical expression to be evaluated by Gmsh.

We can now instruct Gmsh to mesh the defined domain by issuing the following command at the terminal:

```
gmsh -3 manifold.geo
```

This will produce a file manifold.msh which can be opened in Gmsh. This will give you a mesh such as that shown in figure 14.

The full .geo file used in this example is given:

```
//Gmsh .geo file used for spherical manifold.
//
//Applied Modelling and Computation Group
//Last modified 20/10/2011

Point(1) = {0,0,0};            //The center of the sphere
Point(2) = {0, 0,6.37101e+06}; //The North Pole

PolarSphere(1) = {1,2};

//Point 3: 10 lon 5 lat
Point_lon_rad = (10)*(Pi/180);
Point_lat_rad = (5)*(Pi/180);
Point_3_stereographic_x = -Cos(Fabs(Point_lon_rad))*Cos(Fabs(Point_lat_rad))/
    ( 1 + Sin(Fabs(Point_lat_rad)) );
Point_3_stereographic_y = Cos(Fabs(Point_lat_rad))*Sin(Fabs(Point_lon_rad))/
    ( 1 + Sin(Fabs(Point_lat_rad)) );

Point(3) = {Point_3_stereographic_x, Point_3_stereographic_y, 0};

//Point 4: 10 lon 25 lat
Point_lon_rad = (10)*(Pi/180);
Point_lat_rad = (25)*(Pi/180);
Point_4_stereographic_x = -Cos(Fabs(Point_lon_rad))*Cos(Fabs(Point_lat_rad))/
    ( 1 + Sin(Fabs(Point_lat_rad)) );
Point_4_stereographic_y = Cos(Fabs(Point_lat_rad))*Sin(Fabs(Point_lon_rad))/
    ( 1 + Sin(Fabs(Point_lat_rad)) );

Point(4) = {Point_4_stereographic_x, Point_4_stereographic_y, 0};

//Point 5: 30 lon 25 lat
Point_lon_rad = (30)*(Pi/180);
Point_lat_rad = (25)*(Pi/180);
Point_5_stereographic_x = -Cos(Fabs(Point_lon_rad))*Cos(Fabs(Point_lat_rad))/
    ( 1 + Sin(Fabs(Point_lat_rad)) );
Point_5_stereographic_y = Cos(Fabs(Point_lat_rad))*Sin(Fabs(Point_lon_rad))/
```

```
    ( 1 + Sin(Fabs(Point_lat_rad)) );

Point(5) = {Point_5_stereographic_x, Point_5_stereographic_y, 0};

//Point 6: 30 lon 5 lat
Point_lon_rad = (30)*(Pi/180);
Point_lat_rad = (5)*(Pi/180);
Point_6_stereographic_x = -Cos(Fabs(Point_lon_rad))*Cos(Fabs(Point_lat_rad))/
    ( 1 + Sin(Fabs(Point_lat_rad)) );
Point_6_stereographic_y = Cos(Fabs(Point_lat_rad))*Sin(Fabs(Point_lon_rad))/
    ( 1 + Sin(Fabs(Point_lat_rad)) );

Point(6) = {Point_6_stereographic_x, Point_6_stereographic_y, 0};

//Function to break distance between two points into smaller segments.
Function DrawParallel
 alpha = parallelSectionStartingY/parallelSectionStartingX;
 parallelRadius = Sqrt(parallelSectionStartingX^2 + parallelSectionStartingY^2);
 deltaAlpha = (parallelSectionEndingY/parallelSectionEndingX -
    parallelSectionStartingY/parallelSectionStartingX)/pointsOnParallel;
 For t In {1:pointsOnParallel}
  alpha += deltaAlpha;
  newParallelPointX = (parallelSectionStartingX/Fabs(parallelSectionStartingX))
    *parallelRadius/(Sqrt(alpha^2 + 1));
  newParallelPointY = newParallelPointX*alpha;
  newPointOnParallel = newp;
  Point(newPointOnParallel) = {newParallelPointX, newParallelPointY, 0};
 EndFor
 BSpline(newParallelID) = {firstPointOnParallel, newPointOnParallel-
    (pointsOnParallel-1):newPointOnParallel, lastPointOnParallel};
Return

//Draw south-most zonal of the Domain. Point 3 to Point 6

//Assign parameters to variables and then call function DrawParallel,

pointsOnParallel = 200;
parallelSectionStartingX = Point_3_stereographic_x;
parallelSectionStartingY = Point_3_stereographic_y;
firstPointOnParallel = 3;
parallelSectionEndingX = Point_6_stereographic_x;
parallelSectionEndingY = Point_6_stereographic_y;
lastPointOnParallel = 6;
newParallelID = 1;
Call DrawParallel;

//Draw north-most zonal of the Domain. Point 4 to Point 5
//Assign parameters to variables and then call function DrawParallel,
pointsOnParallel = 200;
parallelSectionStartingX = Point_4_stereographic_x;
parallelSectionStartingY = Point_4_stereographic_y;
firstPointOnParallel = 4;
parallelSectionEndingX = Point_5_stereographic_x;
parallelSectionEndingY = Point_5_stereographic_y;
lastPointOnParallel = 5;
newParallelID = 2;
```

```
Call DrawParallel;

//Draw western-most meridional
BSpline(3) = {3, 4};

//Draw eastern-most meridional
BSpline(4) = {5, 6};

Line Loop(6) = {3,2,4,-1};
Plane Surface(7) = {6};
Physical Surface(8) = {7};

Field[50] = MathEval;
Field[50].F = "30000";

Background Field = 50;
```

5 Ocean mesh generation

In this section we briefly present the procedure of mesh generation on domains representative of realistic oceans, including shorelines. Clearly, the shorelines need to be extracted from a relevant source and be reconstructed in the Geometry module of Gmsh. For that purpose, Gmsh features the *GSHHS plugin* (Lambrechts et al., 2008). As its name suggests, this plugin uses the *Global Self-consistent, Hierarchical, High-resolution Shoreline Database*[3] (Wessel and Smith, 1996) as a source of shoreline contours. In specific, the plugin allows the user to select an area on the globe, and then generates a Gmsh geometry script fitting a spline to the shoreline points. The user can then draw open boundaries to form a closed domain.

The interested reader is deferred to http://perso.uclouvain.be/jonathan.lambrechts/gmsh_ocean/ where screen-casts show how to use the plugin and generate meshes featuring higher resolution to towards the shoreline. However, the user can specify plugin options and then invoke plugins from within Gmsh scripts. Thus, a procedure where the user creates a series of scripts and semi-automates the procedure will be presented here in the near future.

[3] available from http://www.ngdc.noaa.gov/mgg/shorelines/gshhs.html

References

Geuzaine, C. and Remacle, J.-F. (2009). 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.

Lambrechts, J., Comblen, R., Legat, V., Geuzaine, C., and Remacle, J.-F. (2008). Multiscale mesh generation on the sphere. *Ocean dynamics*, 58:461–473.

Wessel, P. and Smith, W. H. F. (1996). A global self-consistent, hierarchical, high-resolution shoreline database. *Journal of Geophysical Research*, 101(B4):8741–8743.