

# Mesh<sup>2D</sup> User Manual

S. Rebay, A. Guardone and D. Dussin

*Dipartimento di Ingegneria Aerospaziale  
Politecnico di Milano  
Via La Masa 34, 20158 Milano, Italy  
Email: guardone@aero.polimi.it*

---

Unstructured isotropic mesh generator based on Delaunay algorithm.  
Boundary description, boundary metric enforcement, local domain metric enforcement. An example is showed, including the description of input and output files.

---

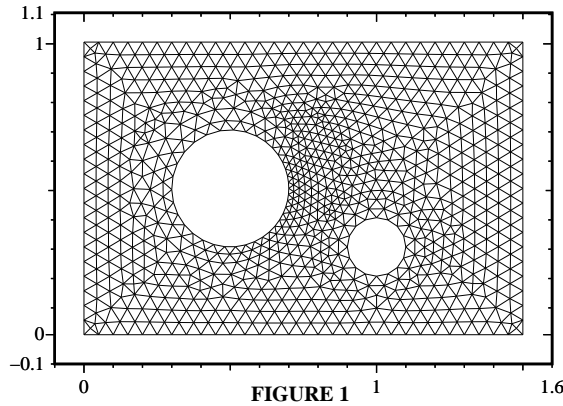
*Key Words:*

## CONTENTS

1. *Introduction.*
2. *Overview of the program.*
3. *Grid generation with Mesh<sup>2D</sup> : an example.*

## 1. INTRODUCTION

$\text{Mesh}^{2D}$  is an unstructured grid generator based on the Delaunay algorithm: its main feature is Rebay's algorithm, which allows to obtain a grid whose elements are almost equilateral (figure 1).



## 2. OVERVIEW OF THE PROGRAM

The program reads the files:

```
boundary.gridname
domain.gridname
```

which must be prepared before running the program and which contain the data necessary to describe the domain to be meshed.

The file `boundary.gridname` has two sections: the first section contains the data for the geometrical description of the boundary, which may be composed of many edges. The second section accounts for the connection of these edges and gathers the requirements for the length of the elements of the boundary, called *boundary metric*.

The file `domain.gridname` allows the imposition of the metric in a discrete number of domain points and the insertion of points without any metric specification.

The program produces two files:

```
knots.gridname, which contains the points of the edges, as specified with the
    geometrical description, and
geometry.gridname, which contains some data about the interpolant curves passing
    through these points.
```

According to the metric requirements, a new number of points is determined along the boundary; these points are connected to each other with a Delaunay based algorithm to form a background grid (which can be viewed plotting `backplot.gridname`); additional grid points (with specified metric) are added and Steiner points (without a metric, which is interpolated over the triangle they belong to) are added.

In most cases the current mesh doesn't match the metric requirements, as it often shows a prevalence of very stretched triangles; the program defines an imaginary boundary to separate the triangles that satisfy the metric requirements from those who do not and inserts a point for every unsatisfactory triangle adjacent to the new boundary.

Corners are splitted and a Laplacian smoothing corrects the points' coordinates by means of a method based on a spring analogy.

The final informations are enclosed in the files:

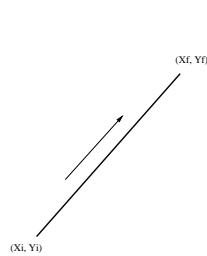
`nodes.gridname`: coordinates of points  
`grid.gridname`: connectivity between elements  
`gridplot.gridname`: informations to plot the elements

### 3. GRID GENERATION WITH Mesh<sup>2D</sup> : AN EXAMPLE

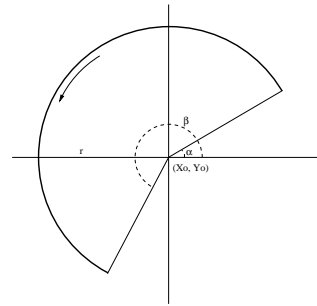
Starting from the current directory it is possible to run the program typing `mesh.exe`; the program asks for the name of the problem, and assumes that the files `boundary.gridname` and `domain.gridname` exist and contain the required records.

#### 3.1. The file `boundary.gridname`

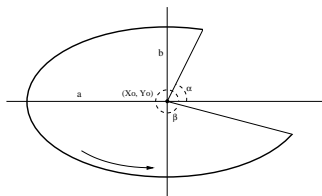
*Geometry input phase:* the boundary consists of a number `nv` of edges and a number `ne` of vertices; every edge begins from a vertex and ends in a vertex; the two vertices coincide for a closed boundary. It is possible to describe an edge by choosing a simple geometry and giving its parameters or referring to a file containing its points; the possible choices for the first case and the corresponding parametres needed are:



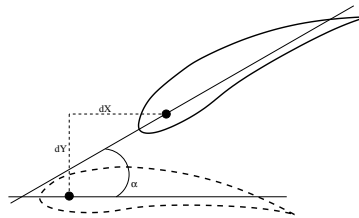
(line)



(circle)



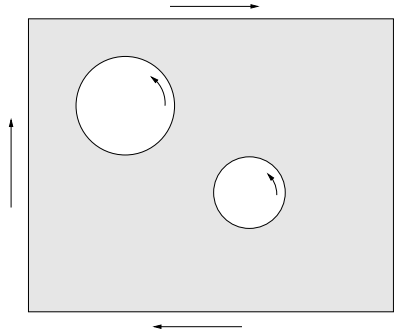
(ellipse)



(data)

- line  
number of points, coordinates of the begin point, coordinates of the end point
- ellipse  
number of points, center coordinates, lengths of the  $x$  and  $y$  semiaxis, begin angle, end angle (degrees)
- circle  
number of points, center coordinates, radius, begin angle, end angle
- data  
name of the file containing the number of points in the first record and, in the following records, the coordinates of points (table 1)  
translation  
coordinates of the displacement vector  
rotation  
rotation angle (degrees)

*\*parameters must be assigned in order to walk along every edge keeping the domain to be triangulated on the right (figure 2).*



**FIGURE 2**

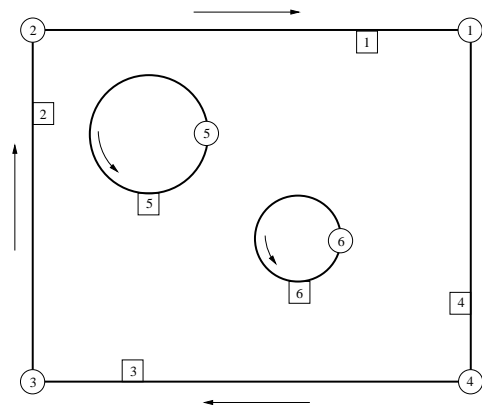
|     |              |               |
|-----|--------------|---------------|
| 100 | 0.1000000000 | 0.0000000000  |
|     | 0.0997986676 | 0.0063423920  |
|     | 0.0991954813 | 0.0126592454  |
|     | :            | :             |
|     | :            | :             |
|     | 0.0991954813 | -0.0126592454 |
|     | 0.0997986676 | -0.0063423920 |
|     | 0.1000000000 | -0.0000000000 |

**TABLE 1**

**The small\_circle file**

*Topology and metric input phase:* this part is necessary to connect the edges defined before and to impose a metric along every edge; this is simply obtained by specifying the length of the triangle's base in a few points identified by the values of a normalized arc length parameter (curvilinear abscissa) relative to the edge they belong to (the growth direction of the abscissa is a consequence of the geometry definition). The metric is then interpolated choosing between three options and the program works in such a way so as to spread the metric information inside the domain and to obtain almost equilateral triangles.

|       |  |
|-------|--|
| nv    | number of vertices   |
| ne    | number of edges  |
| e     | edge number  |
| n     | number of points where the metric is imposed on edge e   |
| bv    | begin vertex for edge e  |
| ev    | end vertex for edge e (the same as bv for a self-closing edge)   |
| idata | metric interpolation over the normalized curvilinear abscissa of the edge e<br>1 linear, 2 sinusoidal, 3 geometrical |
| sdata | normalized curvilinear abscissa for the metric imposition  |
| hdata | length of the boundary element   |



**FIG. 3.** Boundary topology

### 3.2. The file *domain.gridname*

This file contains two necessary parameters:

nstep is the number of times the program have to propose a list of points that are necessary to improve the grid quality and match the metric requirements. If the number indicated is not adequate, a message on the screen will show the percentage of triangles not accepted.

lsmooth allows to include (1) the Laplacian smoothing.

It is possible to insert a list of points together with a specified metric (backgrid points) and a list of points without any explicit metric (Steiner points); in the last case points are inserted interpolating the metric from the background triangle they belong to.

The backgrid and Steiner points sections need the input:

|       |                             |
|-------|-----------------------------|
| n     | number of points            |
| x, y  | point coordinates           |
| hdata | metric (except for Steiner) |

### 3.3. Intermediate output files

*Knots.gridname*: contains the points of every boundary edge.

|        |   |
|--------|---|
| n_curv | number of edges                               |
| dim    | number of dimensions (2)                      |
| points | number of points                              |
| idc    | 0 for line and data, 2 for ellipse and circle |
| x, y   | point coordinates                             |

*Geometry.gridname*: contains the data of the interpolant curve associated with the boundary edge.

|          |                                       |
|----------|---------------------------------------|
| n_curv   | number of edges                       |
| nd       | number of dimensions (2)              |
| ns       | number of curve points                |
| s        | curvilinear abscissa                  |
| x1, x2   | x and y point coordinates             |
| xs1, xs2 | x and y derivatives with respect to s |

### 3.4. Output: grid connectivity and coordinates of points

*Nodes.gridname*: contains domain and boundary points data.

|       |   |
|-------|---|
| np_D  | number of points belonging to the domain (included . . .) |
| np_B  | number of points belonging to the boundary                |
| idx   | domain or boundary point index                            |
| rr    | point coordinates   |
| jd_jb | domain point index corresponding to boundary point index  |
| bound | index of the edge the boundary node belongs to            |

*Grid.gridname*: contains domain and boundary elements data:

|       |   |
|-------|---|
| ne_D  | number of domain elements (included . . .)        |
| ne_B  | number of boundary elements                       |
| idx   | global index of domain or boundary element        |
| type  | element type, 2 = triangle, 1 = segment           |
| j_m   | connectivity matrix, element → nodes              |
| ma_m  | connectivity matrix, element → adjacent elements  |
| bound | index of the edge the boundary element belongs to |

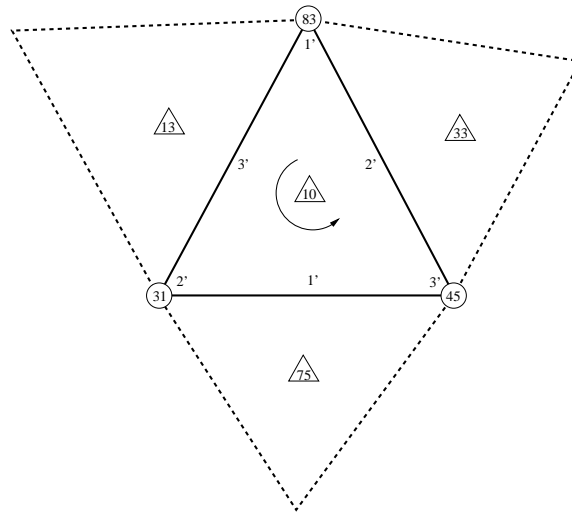


FIG. 4. Global and local indexes

### 3.5. Plot files

*Gridplot.gridname*: contains the data to plot the domain grid.

*Backplot.gridname*: contains the data to plot the background grid.

### 3.6. Echoes on screen

For every boundary edge the program shows the number of points resulting from the imposition of the metric; for the background grid it shows the number of boundary points and segments, included in the following number of domain points and triangles.

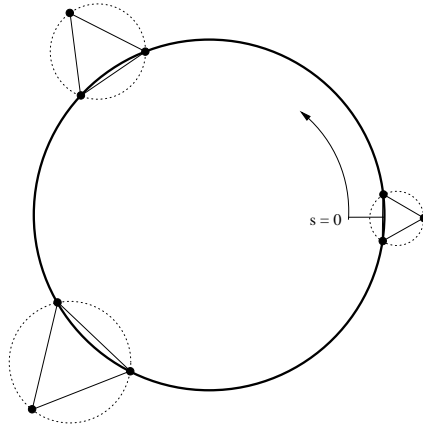
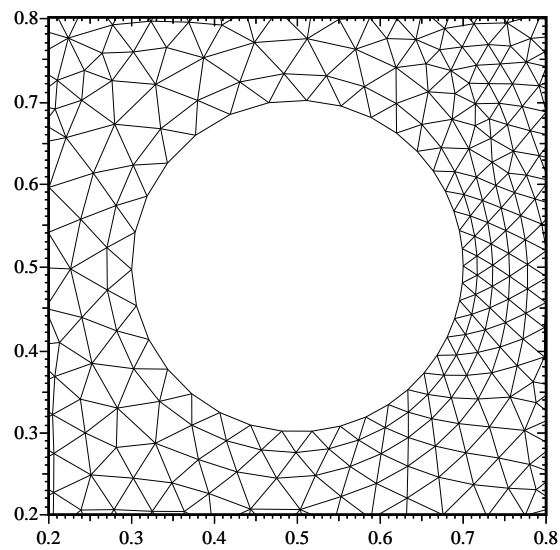
A section is dedicated to the advancing front :

|                         |   |
|-------------------------|---|
| <code>step</code>       | step number                                     |
| <code>nfro</code>       | number of proposed new points                   |
| <code>nadd</code>       | number of added new points                      |
| <code>max_length</code> | the maximum side length between domain elements |

In case the value set for `nstep` is not enough to complete the mesh, the program shows the percentage of triangles that do not satisfy the metric specified.

If the Laplacian smoothing is present, the step number of the iterative method and the residual are showed.

The number of domain triangles is reported as a grid detail by `int_head`.

**FIG. 5.** Boundary metric**FIGURE 6**



```

nv      ne
  6      6
#
##### GEOMETRY#####
#
BEGIN 1 (left line)
line
61 0.0 0.0 0.0 1.0
END
#
BEGIN 2 (up line)
line
61 0.0 1.0 1.5 1.0
END
#
.....
#
BEGIN 5 (circle)
circle
100 0.5 0.5 .2 0 360
END
#
BEGIN 6 (circle)
data
small_circle
  translation
    1.0 0.3
  rotation
    0.0
END
#
##### TOPOLOGY AND METRIC #####
#
BEGIN 1
      e      n      bv      ev
      1      2      1      2
      idata  sdata  hdata
      1      0.0    0.05
      1      1.0    0.05
END
#
BEGIN 2
      e      n      bv      ev
      2      2      2      3
      idata  sdata  hdata
      1      0.0    0.05
      1      1.0    0.05
END
#
.....
#
BEGIN 5
      e      n      bv      ev
      5      4      5      5
      idata  sdata  hdata
      1      0.0    0.02
      1      0.3    0.05
      1      0.7    0.04
      1      1.0    0.02
END
#
BEGIN 6
      e      n      bv      ev
      6      2      6      6
      idata  sdata  hdata
      1      0.0    0.04
      1      1.0    0.04
END

```

**TABLE 2**  
**The topology.circle file**

```

nstep    lsmooth(1/0)
1000      1
#
n (backgrid points)
1
  x        y    hdata
0.80      0.80  0.030
#
n (Steiner points)
1
  x        y
0.50      0.30

```

**TABLE 3**  
**The *domain.circle* file**

```
#####
#   NAME:                circle                                #
#####
#  ne_d   ne_b                                                    #
#    1680   152                                                    #
#####
# ***** DOMAIN *****                                         #
#####
# ++++ ELEMENTS ++++                                           #
#  idx   type                                                    #
#  j_m                                         #
#  ma_m                                         #
#    1       2                                                    #
#   764     1   770                                                #
#   153     0   100                                                #
#    2       2                                                    #
#   770   646   771                                                #
#   154     0   155                                                #
#    3       2                                                    #
#   771   644   772                                                #
#   156     0   154                                                #
#
#   ....   ....
#   ....   ....   ....
#   ....   ....   ....
#
#  1679     2
#    6     17     5
#  1678   1672   1680
#  1680     2
#    7     17     6
#  1679   1674   1677
#####
# ***** BOUNDARY *****                                       #
#####
# ++++ ELEMENTS ++++                                           #
#  idx   type   bound                                           #
#  j_m                                         #
#  ma_m                                         #
#    1       1       1
#    1       2
#    2       0
#    2       1       1
#    3       4
#    3       1
#    3       1       1
#    5       6
#
#   ....   ....   ....
#   ....   ....
#   ....   ....
#
#  151     1       6
#  301    302
#  152    150
#  152     1       6
#  303    304
#    0    151
```

**TABLE 4**  
The *grid.circle* file

```

#####
#   NAME:           circle                                     #
#####
#   nd   np_d  np_b                                     #
#     2   915   304                                     #
#####
# ***** DOMAIN *****                                     #
#####
# ++++  NODES  ++++                                     #
#   idx                                     #
#   rr                                     #
#     1                                     #
# 0.218614037E-01  0.218693044E-01               #
#     2                                     #
# 0.147814262E+01  0.218775883E-01               #
#     3                                     #
# 0.147810419E+01  0.978162969E+00               #
#     4                                     #
# 0.218773470E-01  0.978142850E+00               #
#     5                                     #
# 0.878989450E+00  0.687758689E+00               #
#                                     #
# ....                                     #
# .....                                     #
#                                     #
#     914                                     #
# 0.107071034E+01  0.229289218E+00               #
#     915                                     #
# 0.109239125E+01  0.261730356E+00               #
#####
# ***** BOUNDARY *****                                     #
#####
# ++++  NODES  ++++                                     #
#   idx  jd_jb  bound                                     #
#     1    764    1                                     #
#     2    770    1                                     #
#     3    770    1                                     #
#     4    771    1                                     #
#     5    771    1                                     #
#     6    772    1                                     #
#                                     #
# ....    ....    ....                                     #
#                                     #
#    303    915    6                                     #
#    304    769    6                                     #

```

**TABLE 5**  
**The *nodes.circle* file**