Jednoocí gmsh

GMSH - A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities
general info/downloads: **http://gmsh.info/**
documentation: **http://gmsh.info/doc/texinfo/gmsh.pdf**

Key points:

- Geometry: geometrical entity definition Gmsh uses a boundary representation ("BRep") to describe geometries. Models are created in a bottom-up flow by successively defining points, oriented curves (line segments, circles, ellipses, splines, . . .), oriented surfaces (plane surfaces, surfaces, triangulated surfaces, . . .) and volumes. Groups of geometrical entities (called "physical groups") can also be defined, based on these elementary geometric entities. Gmsh's scripting language allows all geometrical entities to be fully parameterized.

- Mesh: A finite element mesh is a tessellation of a given subset of the three-dimensional space by elementary geometrical elements of various shapes (in Gmsh's case: lines, triangles, quadrangles, tetrahedra, prisms, hexahedra and pyramids), arranged in such a way that if two of them intersect, they do so along a face, an edge or a node, and never otherwise.

  All the finite element meshes produced by Gmsh are considered as "unstructured", even if they were generated in a "structured" way (e.g., by extrusion). This implies that the elementary geometrical elements are defined only by an ordered list of their nodes but that no predefined order relation is assumed between any two elements

  The mesh generation is performed in the same bottom-up flow as the geometry creation: curves are discretized first; the mesh of the curves is then used to mesh the surfaces; then the mesh of the surfaces is used to mesh the volumes. In this process, the mesh of an entity is only constrained by the mesh of its boundary. For example, in three dimensions, the triangles discretizing a surface will be forced to be faces of tetrahedra in the final 3D mesh only if the surface is part of the boundary of a volume; the line elements discretizing a curve will be forced to be edges of tetrahedra in the final 3D mesh only if the curve is part of the boundary of a surface, itself part of the boundary of a volume; a single node discretizing a point in the middle of a volume will be forced to be a node of one of the tetrahedra in the final 3D mesh only if this point is connected to a curve, itself part of the boundary of a surface, itself part of the boundary of a volume.

Interactive mode using the GUI:

Examples of simple .geo files
t1.geo, t2.geo, ... (launch: gmsh t1.geo)

Combination of multiple files (background view): gmsh t1.geo view1.pos view5.msh

Non-interactive mode, scripting:

Examples: gmsh t1.geo -2
enceladus demo

comments, Expressions floating point/string, operators, built-in functions, user defined macros, loops and conditionals, general commands.

Geometry module:

Logic: points → curves → surfaces → volumes (elementary geometrical entities)
each elementary geometric entity must be tagged
manipulations with elementary geometric entities: Translate, Rotate, Scale, Symmetry, Extrusions

Mesh module:
currently 3 2D unstructured algorithms, 2 3D unstructured algorithms
2D: init:2D Delaunay triangulation via divide-and-conquer algorithm
then MeshAdapt, Delaunay, Frontal

Specifying mesh element sizes:

- Mesh.CharacteristicLengthFromPoints (default) - specify desired mesh element sizes at the geometrical points of the model (with the Point command) example
- Mesh.CharacteristicLengthFromCurvature is set (it is not by default), the mesh will be adapted with respect to the curvature of the geometrical entities example
- Mesh size “fields” , examples: Attractor, Threshold, …..
- …..

Examples of export to FEniCS via doflin-convert: crater0,crater1, crater2