FacetModeller: Software for manual creation, manipulation and analysis of 3D surface-based models

Peter G. Lelièvre[a,*], Angela E. Carter-McAuslan[a], Michael W. Dunham[a], Drew J. Jones[a], Mariella Nalepa[a], Chelsea L. Squires[a], Cassandra J. Tycholiz[a], Marc A. Vallée[a], Colin G. Farquharson[a]

[a]*Memorial University of Newfoundland, St. John's, NL, Canada*

Abstract

The creation of 3D models is commonplace in many disciplines. Models are often built from a collection of tessellated surfaces. To apply numerical methods to such models it is often necessary to generate a mesh of space-filling elements that conforms to the model surfaces. While there are meshing algorithms that can do so, they place restrictive requirements on the surface-based models that are rarely met by existing 3D model building software. Hence, we have developed a Java application named FacetModeller, designed for efficient manual creation, modification and analysis of 3D surface-based models destined for use in numerical modelling.

Keywords: 3D model, geological model, meshing, manual model building, numerical methods, Java

1. Motivation and significance

The creation of 3D models for visualization or quantitative analysis is commonplace in many disciplines. For example, in applied geophysics, 3D models are often created, modified and queried throughout the life of an exploration project to determine the nature and composition of the Earth volume of interest (VOI). Models are often built from a collection of surfaces that comprise tessellated triangles or other planar polygonal shapes. For example, in geological models, these surfaces define the contacts between different rock units; refer to [1] for a discussion of some of the techniques typically used to generate those surfaces. We refer to the tessellated triangles or polygons as "facets" and these types of models as "surface-based" models.

*Corresponding author
Email address: `plelievre@mun.ca` (Peter G. Lelièvre)

Computer modelling in many disciplines involves numerical calculation of physical phenomena that are affected by spatial variations in the physical properties in the VOI. Many numerical methods discretize the VOI on a mesh of space-filling elements, often referred to as mesh "cells". For example, a 3D rectilinear mesh comprises rectangular prisms arranged in a structured grid; a 3D unstructured tetrahedral mesh comprises tetrahedra. To apply numerical methods to surface-based models it is therefore often necessary to generate a mesh that conforms to the model surfaces. For example, in applied geophysics, 3D surface-based geological models may be used to help constrain geophysical forward and inverse modelling performed on conforming meshes [e.g. 2, 3, 4, 5, 6, 7, 8].

Surface-based models can be fed into meshing algorithms that generate unstructured meshes. For example, the triangular facets in a surface-based model become the faces of the tetrahedra in a mesh: see Fig. 1. There are many mesh generation software packages for doing so [e.g. 9, 10, 11] and for improving such meshes [e.g. 12]. For most mesh generation methods, the input must be a piecewise linear complex (PLC), a concept first introduced by [13], representing the 3D domain to be meshed.

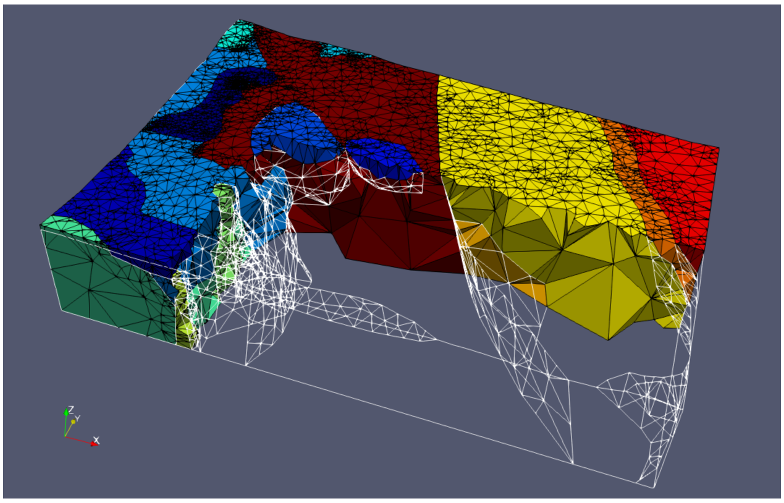

Figure 1: A mesh of tetrahedral cells that conforms to a surface-based model (a collection of tessellated triangles). The edges of the facets in the surface-based model are white, the mesh tetrahedra are coloured such that each different region in the model has a different colour, and the edges of the mesh tetrahedra are black. Some of the tetrahedral cells have been removed from the southeast of the mesh to expose the surfaces lying within the mesh. This figure is a screenshot from ParaView [14].

Reference [15] provides the requirements that a PLC must satisfy, which we paraphrase here. A PLC is a set of planar polygonal facets that satisfies the following properties:

- the boundary of every facet (an edge) is a union of facets
- if two distinct facets intersect then their intersection is a union of facets.

These requirements are demonstrated visually in Fig. 2. We use the term “node” to denote the vertices of the facets in a PLC, and we use “edges” to denote line-segments that connect pairs of nodes, that is, the edges of the facets.

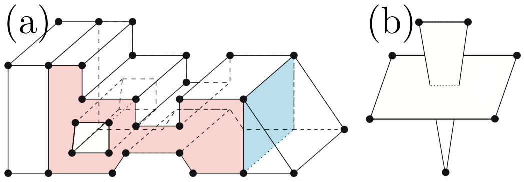

Figure 2: (a) A 3D piecewise linear complex: the pink shaded facet is on the boundary of the domain and has a hole in it; the blue shaded facet is an interior polygon separating two sub-domains. (b) A configuration of two facets that breaks the rules of a valid PLC because there is no edge at the intersection. Nodes are drawn as black dots and edge elements as black lines. Adapted from [15].

The above requirements on a PLC must apply to any surface-based model with which we wish to generate a conforming mesh for performing numerical computations. For such computations, further requirements may exist related to the “quality” of the mesh derived from the PLC. For example, when finite volume or finite element methods are used to simulate physical phenomena, numerical modelling accuracy and solution times can depend critically on the quality of the unstructured mesh [e.g. 16, 17, 18, 19, 20, 21]. Obtaining an acceptable mesh quality may only be possible if the input PLC is itself of a high enough quality.

The definition of mesh quality depends on the intended application and numerical methods employed but is generally related to the geometry of the tetrahedral mesh cells [22]. A general guideline is that tetrahedral cells with very small or large dihedral angles should be avoided. Hence, similar considerations apply to the input PLC: triangular facets with very small or large vertex angles should be avoided.

There are many software packages that are capable of building 3D surface-based models but to date there are none that have fully met our needs for generating quality PLCs for use in our geophysical computations [e.g. 17, 18, 19, 20]. Often, such software uses automated procedures that may, for example, interpolate curves and surfaces between sparse measured data, and automatically generate the triangular tessellations on the interpolated surfaces. Such automated processing reduces the model building time but can introduce some unwanted artifacts, even if there is some user control on the automated processes.

Recent research is attempting to ameliorate some of these issues through numerical procedures [e.g. 23]. However, automated model building tools may provide less-than optimal results and workflows must then be devised using different model building and mesh generation software [e.g. 24]. Such workflows can only benefit from the availability of software that eases the development of such workflows [e.g. 25, 26] and provides helpful tools for manual model building tasks.

2. Software description

2.1. Software Architecture

FacetModeller is a Java application designed for efficient manual creation, modification and analysis of quality 3D PLCs, for example geological surface-based models destined for use in geophysical numerical modelling. FacetModeller is not intended to be a replacement for other 3D model building software packages: the intention is not to duplicate the automatic model building tasks that such packages can perform , but instead provide an efficient tool for manual model building tasks. FacetModeller is designed to help users honour the requirements of a valid PLC and control the quality of their surface-based models.

3D surface-based models in FacetModeller comprise the following components:

- Different parts of the model can be assigned to different "sections". These may be: concrete, spatially-connected objects, i.e. planar slices through the 3D model; or abstract containers for organizing different pieces of information. Each section may or may not have an image associated with it.
- The basic building blocks of a surface-based model are nodes and facets. Nodes are attached to different sections. Facets are specific connections between nodes.
- Different types of nodes and facets are distinguished by assigning them to different "groups". Each group has user-defined drawing colours associated with it. The concepts of sections and groups can be used to help organize the different parts of a model.

There are two main data input types for FacetModeller:

- images that must be digitized; for example, geological maps, interpolated vertical cross sections, or interpolated horizontal depth sections
- pre-defined 3D model surfaces, where each surface may be a tessellation of facets or a collection of unconnected nodes lying on the surface.

Fig. 3 shows the FacetModeller GUI. As FacetModeller is a Java program, the GUI may look different on different operating systems. The FacetModeller window is split into three main panels. On the left are various buttons and selection boxes that allow users to select which objects they wish to display and work with. In the centre of the GUI is a 2D viewing panel and on the right is a 3D viewing panel. In the 2D viewing panel, users can define nodes and facets via cursor interactions. The 3D viewing panel is only used for viewing: no model building occurs directly through user interaction with the 3D viewing panel. The 2D viewing panel displays the currently selected section image and any nodes or facets selected for display. In the 2D viewing panel, all objects specified for drawing are projected onto the plane of the current section. Alternatively, the projection of the 2D viewing panel can be made to mirror that of the 3D view. Various buttons appear below and above the two viewing panels to allow the user to zoom and access common menu items.

2.2. Software Functionalities

Although we focus on 3D models in this paper, FacetModeller can also be used for building 2D models. For 2D models, there is only a single section and facets are no longer triangles but line-segment elements that connect pairs of nodes. There is a reduced tool set required for building 2D models and hence the 2D version of the GUI is somewhat simplified from that seen in the figures presented in this paper.

There are several cursor interaction modes, or "click modes", available for interacting with the 2D viewing panel. These click modes allow the user to perform various model building tasks by clicking or dragging with the cursor on the 2D viewing panel. Here we provide a complete list of the tasks that users are able to perform with those interactions:

- set the origin of the 2D and 3D viewing panels
- spatially register a section image
- obtain information about the different model components
- add nodes at specific locations on spatially registered section images
- add nodes on a facet edge or anywhere on the plane of a facet
- delete, move and merge nodes
- define triangular facets, or polygonal facets with more than three nodes
- delete facets

- reverse the node order in facets, which is helpful for numerical modelling methods that might provide different results depending on the order [e.g. 27]
- change the group and sections associated with different nodes or facets
- perform edge-flip operations to improve model quality.

Some of those tasks require a single click while others require multiple clicks, and some support click-and-drag interaction. For some tasks, temporary overlays are drawn in the 2D or 3D viewing panels; for example, to indicate objects currently being worked with or a candidate object about to be defined. The drawing colours for those overlays, and for any model components, can be changed as desired by the user.

FacetModeller allows users to obtain various information about the model, via dialogs or through cursor interaction, including:

- the number of nodes and facets for the entire model, for each group and for each section
- the 3D spatial coordinates for a specific node
- the number of facets associated with a specific node
- the node indices associated with a specific facet
- the order of the nodes in a specific facet
- the minimum vertex angle in a specific facet.

FacetModeller also allows users to highlight specific nodes and facets by their index (order listed in memory), which can be helpful for cross-referencing against lines in imported or exported ASCII files.

FacetModeller's model cleaning tools provide users the ability to identify and delete the following:

- facets with zero area
- facets with duplicated nodes
- facets with fewer than three unique nodes
- non-planar facets
- duplicated facets
- duplicated nodes.

The current version of FacetModeller does not detect intersecting facets that break the requirements of a valid PLC (see Fig. 2(a)). Instead, we rely on TetGen [11] for that task, which can identify the indices of offending facets such that they can be easily identified in FacetModeller.

The model can be saved in various ASCII formats:

- .node, .ele and .poly files (refer to the TetGen documentation [11] for those file formats)

- .vtu files for visualization, for example in ParaView [14]
- "FacetModeller session" files that enable users to save their work and restart their model building session at later times.

3. Illustrative Examples

Example 1: Fig. 3 shows a geological model built in FacetModeller for teaching purposes. The geological scenario involves sedimentary layers that have been folded into a synform plunging to the south-east and then cross-cut by igneous units. The model was built from a geological map and interpreted geological sections (planar slices through a 3D Earth). Each section image was first imported into FacetModeller, spatially registered, and digitized by defining nodes and facets attached to it. Intersecting surfaces were carefully sewn together, to honour the requirements of a valid PLC, using various tools available in FacetModeller.

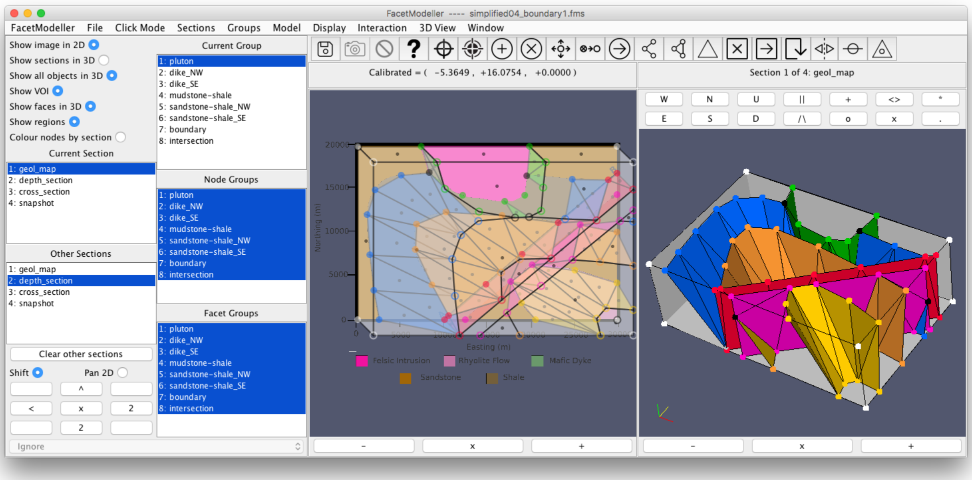

Figure 3: The final model for Example 1, with all nodes and facets defined. The 2D viewing panel shows an overhead view (north up the page); the geological map is displayed with model component overlays; the nodes on the depth section (drawn as unfilled circles) have been shifted to the south-east to aid visualization. The 3D viewing panel shows an elevated view roughly from the south-east; some of the facets around the VOI boundary (light grey) are not shown because they would obscure the other surfaces. In both viewing panels, the nodes and facet patches are drawn in their designated colours and facet edges are drawn black. The different coloured nodes and facets correspond to the different groups they have been assigned to.

Example 2: Fig. 4 shows a more complicated geological model of the core of a porphyry copper system including different geological phases. This model was used for numerical modelling purposes. Initially, several surfaces were built using other software packages: Gocad by Paradigm [28] and Autodesk Meshmixer [29]. Those surfaces were imported into FacetModeller as pre-defined tessellated surfaces. The surfaces nearly made contact with each other along their boundaries but they had been built independently and were not sewn together following the requirements of a valid PLC. Hence, that was the major task to which FacetModeller was applied for this example. There are a total of seventeen tessellated surfaces for this model, including the topography surface. The model in Fig. 4 represents a valid PLC for providing to a meshing program, and Fig. 1 shows a mesh generated from the PLC.

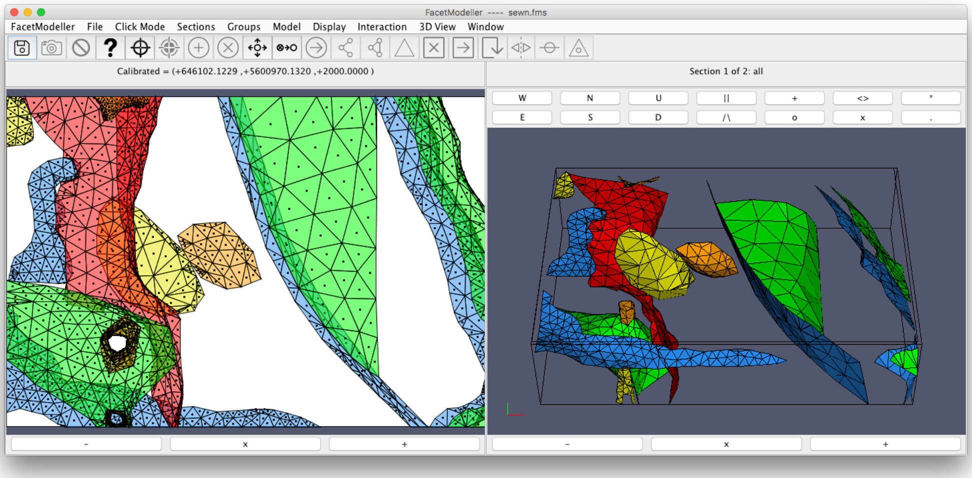

Figure 4: The final model for Example 2, with all nodes and facets defined. The 2D viewing panel shows an overhead view (north up the page). The 3D viewing panel shows an elevated view from the south. In both viewing panels, facets around the VOI boundary, including the topography surface, are hidden so as not to obscure the other model surfaces. The tool panel has been hidden in the FacetModeller GUI to maximize the width of the viewing panels.

4. Major Assets

FacetModeller contains various features that enable users to create and manipulate 3D models efficiently. One of the most helpful features for model creation is the way in which triangular facets are generated. As the user moves the cursor around in the 2D viewing panel, candidate facets are displayed. FacetModeller automatically determines all the possible candidate triangular facet definitions within some tuneable neighbourhood around the cursor and displays the facet with centroid closest to the cursor. Often, it is possible to build the model with different triangular tessellations with different quality characteristics; to aid in the generation of quality PLCs, FacetModeller displays the smallest vertex angle of the candidate facet in a text bar above the 2D viewing panels. Once the desired candidate facet is found, the user generates that facet by clicking the mouse. The new facet is then added to the model and drawn using the colour specified for its group. This approach greatly speeds up the model building process.

The ability to fix intersecting surfaces such that they honour the requirements of a valid PLC is one of the main strengths of FacetModeller and fills one of the major gaps of other currently existing software packages for 3D model building. As an example, Fig. 5 shows two of the several surfaces for Example 1 that needed to be sewn together to create the final model. Nodes along the intersection are drawn red. FacetModeller provides several tools to help with this sewing task. One approach is simply to delete existing facets along an intersection and generate new ones as required. This may sometimes require that new nodes be created at various locations on an existing surface. Another option is to merge nodes together across the intersection and then delete any zero-area facets generated through that merging.

Another important aspect of FacetModeller is that it is designed to help improve the quality of PLCs. FacetModeller provides several tools to help with this task:

- new nodes can be inserted at any location on an existing surface and incorporated into the surface to improve quality
- nodes can be moved across a surface
- edge-flip operations can be performed, similar to those involved when creating Delaunay triangulations [e.g. 30]
- the minimum vertex angle in a model facet can be shown during creation or queried afterwards.

FacetModeller provides some visualization options that aid the process of manual model creation and manipulation. In the 2D viewing panel, facets are drawn as transparent coloured patches with black edges: this allows all facets to remain visible regardless of the projection used for the 2D viewing panel. In the 3D viewing panel, facets are drawn non-transparent so that facets closer to the viewing camera obscure those behind them: this provides a more realistic and intuitive 3D viewing experience but can require users to rotate the 3D view such that the new facets being defined are visible. Having these two different drawing choices (transparent vs non-transparent facets), and having two different but simultaneously available views, are both helpful

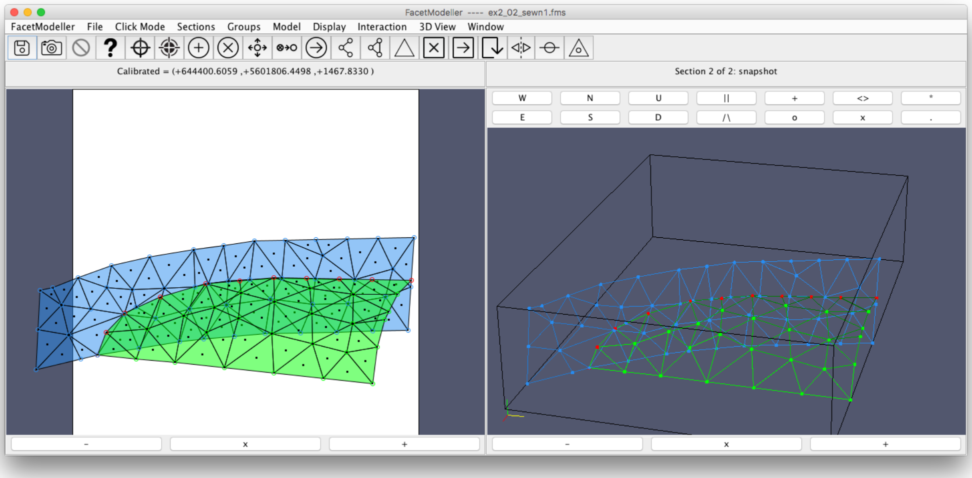

Figure 5: Part-way through the model building process for Example 2. Two surfaces (blue and green) have been loaded into FacetModeller. Where the two surfaces intersect, the nodes on the green surface have been moved to a temporary group and drawn red. The projection of the 2D viewing panel is set to mirror the 3D view: an elevated view roughly from the north-east. In the 2D panel, the nodes and facet patches are drawn in their designated colours and facet edges are drawn black; in the 3D panel, the nodes and the edges of the surface facets are drawn in their designated colours. The tool panel has been hidden in the FacetModeller GUI.

for many manual model building tasks because they improve visualization of the model being built.

5. Impact

FacetModeller has been used in several studies in which complicated geological models were generated for numerical modelling work on unstructured meshes [e.g. 8, 31, 32, 33, 34, 35, 36]. Without FacetModeller, those studies would not have been possible.

FacetModeller should also be of interest to those working on rectilinear meshes. More complicated 3D models can be built in FacetModeller using an unstructured surface-based representation before interpolating the model onto a rectilinear mesh to generate a pixellated approximation [e.g. 2, 6].

Although we have designed FacetModeller for building geological surface-based models, the software may be helpful in other fields. For example, [37] developed a software package they named JMorph, for performing morphometric measurements on digital images of fossil assemblages. JMorph makes measurements in what is essentially a 2D environment, the 2D data being a specific digital image. 3D data for morphometric applications might correspond to point clouds from laser scanning or stacked section images from computerized tomography (CT) scanning. That information could easily be imported into FacetModeller, which could thereby act as a 3D extension to JMorph, provided that new click modes were developed for the required 3D morphometric measurements. FacetModeller may also be helpful for engineering problems where 3D models must be built, for example limit analysis of structures like vaults, arches and complex masonry geometries [e.g. 38, 39].

While FacetModeller works with explicit representations of surfaces, some models are built using an underlying control surface. For example, non-uniform rational basis spline (NURBS) surfaces [e.g. 40] and subdivision surfaces [e.g. 41]. FacetModeller can not deal with spline surfaces like NURBS. However, a coarse model could be created in FacetModeller and refined using surface subdivision.

6. Conclusions

We have developed a new software tool, FacetModeller, that is custom-built for efficient manual creation, manipulation and analysis of 3D surface-based models. FacetModeller's three major assets are:

1. the speed in which manual model building tasks can be performed using carefully designed cursor interactions

2. functionality that helps users honour the requirements of a valid PLC, and improve the quality of their PLC
3. providing two simultaneous views of the model being built, each view providing different but complimentary information to the user.

These assets are important when manual model building is the only viable option, for example when automatic methods provide inappropriate results such as poor quality tessellations or intersecting facets.

FacetModeller is written in Java and will therefore run on any operating system that can run Java, including Linux, Mac and Windows. FacetModeller is available as source code or a compiled JAR file. The code has been developed with extensibility in mind: developers can define new click modes using the ClickTask interface (a Java interface). Other data processing tools could also be added to FacetModeller through its modular design, which follows the model–view–controller software architecture pattern.

FacetModeller should be of use to anyone wishing to build 3D surface-based models, particularly for numerical modelling work on unstructured meshes. We hope that FacetModeller will be of use to other researchers undertaking 3D numerical modelling studies, in geophysics or other fields.

[1] G. Caumon, P. Collon-Drouaillet, C. Le Carlier de Veslud, S. Viseur, J. Sausse, Surface-based 3D modeling of geological structures, Mathematical Geosciences 41 (2009) 927–945.

[2] N. Phillips, D. Oldenburg, J. Chen, Y. Li, P. Routh, Cost effectiveness of geophysical inversions in mineral exploration: applications at San Nicolas, The Leading Edge 20 (12) (2001) 1351–1360, doi:10.1190/1.1487264.

[3] P. K. Fullagar, G. Pears, D. Hutton, A. Thompson, 3D gravity and aeromagnetic inversion for MVT lead-zinc exploration at Pillara, Western Australia, Exploration Geophysics 35 (2) (2004) 142–146, doi:10.1071/EG04142.

[4] J. McGaughey, The common earth model: A revolution in mineral exploration data integration, in: J. Harris (Ed.), GIS For the Earth Sciences, Vol. 44, Geological Association of Canada Special Publication, 2006, Ch. 25, pp. 567–576.

[5] J. McGaughey, Geological models, rock properties, and the 3D inversion of geophysical data, in: B. Milkereit (Ed.), Proceedings of Exploration 07: Fifth Decennial International Conference on Mineral Exploration, 2007, pp. 473–483.

[6] C. G. Farquharson, M. R. Ash, H. G. Miller, Geologically constrained gravity inversion for the Voisey's Bay ovoid deposit, The Leading Edge 27 (1) (2008) 64–69.

[7] A. Guillen, P. Calcagno, G. Courrioux, A. Joly, P. Ledru, Geological modelling from field data and geological knowledge Part II. Modelling validation using gravity and magnetic data inversion, Physics of the Earth and Planetary Interiors 171 (2008) 158–169.

[8] P. G. Lelièvre, C. G. Farquharson, Gradient and smoothness regularization operators for geophysical inversion on unstructured meshes, Geophysical Journal International 195 (1) (2013) 330–341, doi:10.1093/gji/ggt255.

[9] The CGAL Project, CGAL: Computational Geometry Algorithms Library, https://www.cgal.org, last accessed November 2017. (2017).

[10] Los Alamos National Laboratory, LaGriT: Los Alamos Grid Toolbox, http://lagrit.lanl.gov, last accessed November 2017. (2017).

[11] H. Si, TetGen: a quality tetrahedral mesh generator and a 3D delaunay triangulator, http://wias-berlin.de/software/tetgen/, last accessed November 2017. (2017).

[12] B. M. Klingner, J. R. Shewchuk, Stellar: A tetrahedral mesh improvement program, https://people.eecs.berkeley.edu/~jrs/stellar/, last accessed November 2017. (2017).

[13] G. L. Miller, D. Talmor, S. Teng, N. Walkington, H. Wang, Control volume meshes using sphere packing: Generation, refinement and coarsening, Proceedings of the 5th International Meshing Roundtable, Sandia National Laboratories (1996) 47–61.

[14] Kitware, Inc., Los Alamos National Laboratory, ParaView: an open-source, multi-platform data analysis and visualization application, https://www.paraview.org, last accessed November 2017. (2017).

[15] H. Si, TetGen, a delaunay-based quality tetrahedral mesh generator, ACM Trans. on Mathematical Software 41 (2).

[16] C. Rücker, T. Günther, K. Spitzer, Three-dimensional modelling and inversion of DC resistivity data incorporating topography – I. modelling, Geophysical Journal International 166 (2) (2006) 495–505, doi:10.1111/j.1365–246X.2006.03010.x.

[17] P. G. Lelièvre, C. G. Farquharson, C. A. Hurich, Computing first-arrival seismic traveltimes on unstructured 3D tetrahedral grids using the fast marching method, Geophysical Journal International 184 (2) (2011) 885–896.

[18] H. Jahandari, C. Farquharson, Forward modeling of gravity data using finite-volume and finite-element methods on unstructured grids, Geophysics 78 (2013) G69–G80, doi:10.1190/geo2012–0246.1.

[19] S. Ansari, C. G. Farquharson, 3D finite-element forward modeling of electromagnetic data using vector and scalar potentials and unstructured grids, Geophysics 79 (2014) E149–E165, doi:10.1190/geo2013–0172.1.

[20] H. Jahandari, C. Farquharson, A finite-volume solution to the geophysical electromagnetic forward problem using unstructured grids, Geophysics 79 (6) (2014) E287–E302, doi:10.1190/geo2013–0312.1.

[21] H. Cai, X. Hu, J. Li, M. Endo, B. Xiong, Parallelized 3D CSEM modeling using edge-based finite element with total field formulation and unstructured mesh, Computers and Geosciences 99 (2017) 125–134.

[22] J. R. Shewchuk, What is a good linear element? Interpolation, conditioning, and quality measures, Proceedings of the 11th International Meshing Roundtable, Sandia National Laboratories (2002) 115–126.

[23] J. Pellerin, B. Lévy, G. Caumon, A. Botella, Automatic surface remeshing of 3D structural models at specified resolution: A method based on voronoi diagrams, Computers and Geosciences 62 (2014) 103–116.

[24] B. Zehner, J. H. Börner, I. Görz, K. Spitzer, Workflows for generating tetrahedral meshes for finite element simulations on complex geological structures, Computers and Geosciences 79 (2015) 105–117.

[25] GeoRessources, CREGU, RINGMesh: A free programming library for geological model meshes, http://www.ring-team.org/software/ringmesh, last accessed November 2017. (2017).

[26] J. Pellerin, A. Botella, F. Bonneau, A. Mazuyer, B. Chauvin, B. Lévy, G. Caumon, RINGMesh: A programming library for developing mesh-based geomodeling applications, Computers and Geosciences 104 (2017) 93–100.

[27] M. Okabe, Analytic expressions for gravity anomalies due to homogeneous polyhedral bodies and translations into magnetic anomalies, Geophysics 44 (4) (1979) 730–741.

[28] J. L. Mallet, Gocad: A computer aided design program for geological applications, in: A. K. Turner (Ed.), Three-Dimensional Modeling with Geoscientific Information Systems, Vol. 354 of NATO ASI Series (Series C: Mathematical and Physical Sciences), Springer, Dordrecht, 1992, pp. 123–141.

[29] Autodesk, Autodesk Meshmixer, http://www.meshmixer.com, last accessed November 2017. (2017).

[30] J. R. Shewchuk, Triangle: engineering a 2D quality mesh generator and delaunay triangulator, in: M. C. Lin, D. Manocha (Eds.), Applied Computational Geometry: Towards Geometric Engineering, Vol. 1148 of Lecture Notes in Computer Science, Springer-Verlag, Berlin, Germany, 1996, pp. 203–222, DOI:10.1007/BFb0014497.

[31] A. Carter-McAuslan, P. G. Lelièvre, C. G. Farquharson, A study of fuzzy c-means coupling for joint inversion, using seismic tomography and gravity data test scenarios, Geophysics 80 (1) (2015) W1–W15, doi:10.1190/geo2014–0056.1.

[32] M. Darijani, C. G. Farquharson, Synthetic modeling and joint inversion of gravity and seismic refraction data for overburden stripping in the Athabasca Basin, Canada, in: SEG Technical Program Expanded Abstracts 2016, Dallas, 2016, pp. 5011–5016, doi:10.1190/segam2016–13947873.1.

[33] M. Dunham, S. Ansari, C. G. Farquharson, Application of 3D marine controlled-source electromagnetic finite-element forward modeling to hydrocarbon exploration in the Flemish Pass Basin offshore Newfoundland, Canada, Geophysics 83 (2) (2018) WB33–WB49, doi:10.1190/GEO2017–0451.1.

[34] H. Jahandari, C. G. Farquharson, 3D minimum-structure inversion of magnetotelluric data using the finite-element method and tetrahedral grids, in: SEG Technical Program Expanded Abstracts 2016, Dallas, 2016, pp. 998–1003, doi:10.1190/segam2016–13866304.1.

[35] D. Jones, S. Ansari, C. G. Farquharson, Synthesizing time-domain electromagnetic data for graphitic fault zones and associated uranium deposits in the Athabasca Basin, Canada, in: SEG Technical Program Expanded Abstracts 2016, Dallas, 2016, pp. 2206–2210, doi:10.1190/segam2016–13866371.1.

[36] M. Nalepa, S. Ansari, C. G. Farquharson, Finite-element simulation of 3D CSEM data on unstructured meshes: An example from the east coast of canada, in: SEG Technical Program Expanded Abstracts 2016, Dallas, 2016, pp. 1048–1052, doi:10.1190/segam2016–13949192.1.

[37] P. G. Lelièvre, M. Grey, JMorph: Software for performing rapid morphometric measurements on digital images of fossil assemblages, Computers and Geosciences 105 (2017) 120–128.

[38] E. Milani, G. Milani, A. Tralli, Limit analysis of masonry vaults by means of curved shell finite elements and homogenization, International Journal of Solids and Structures 45 (2008) 5258–5288, doi:10.1016/j.ijsolstr.2008.05.019.

[39] G. Milani, Upper bound sequential linear programming mesh adaptation scheme for collapse analysis of masonry vaults, Advances in Engineering Software 79 (2015) 91–110, doi:10.1016/j.advengsoft.2014.09.004.

[40] L. Piegl, W. Tiller, The NURBS Book, Monographs in Visual Communication, Springer-Verlag Berlin Heidelberg, 1997.

[41] J. Peters, U. Reif, Subdivision Surfaces, Vol. 3 of Geometry and Computing, Springer-Verlag Berlin Heidelberg, 2008.

Required Metadata

Current code version

Nr.	Code metadata description	Please fill in this column
C1	Current code version	3.0
C2	Permanent link to code/repository used for this code version	github.com/pglelievre/facetmodeller
C3	Legal Code License	GNU General Public License (GPL)
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	Java
C6	Compilation requirements, operating environments & dependencies	The github link above contains a NetBeans project FacetModeller which uses MyLibrary (also supplied at github link above)
C7	If available Link to developer documentation/manual	*will be made available on github*
C8	Support email for questions	plelievre@mun.ca

Table 1: Code metadata (mandatory)