Pillargrid Conversion for Seismic Modeling

Charles Cote, Michael Prange, Hugues Djikpesse* (Schlumberger)

This paper was prepared for presentation during the 12th International Congress of the Brazilian Geophysical Society held in Rio de Janeiro, Brazil, August 15-18, 2011.

This paper is a short version of a more complete article submitted in March 2011 for publication in Geophysics.

Abstract

Pillargrid and unstructured mesh models describe geometry and properties in different ways that are optimized for distinct types of simulation. For instance, finite-element simulators typically require unstructured, watertight and piecewise-polynomial descriptions of geometry and properties, while industry-standard fluid-flow simulators often use a pillargrid description of "connected bricks" in which the hexahedral cells may have voids between them when converted into a piecewise-linear description of the surface geometries, and the properties within each cell are constant.

A conversion method is described to extract the geometry and properties from a pillargrid to build a watertight, piecewise linear model. The algorithm comprises partitioning the pillargrid into subvolumes within which properties are smoothly varying, building triangulated bounding surfaces for each of the subvolumes, and generating a mesh of property nodes for each of the subvolumes. Such conversion is useful for instance to simulate geophysical and geomechanical wave fields inside high-resolution reservoir models that are consistent with fluid-flow simulation models. The proposed algorithm has been validated on both synthetic and field examples.

Introduction

Two types of models are traditionally used for representing geological structures in the oil and gas industry: pillargrid models (also known as corner-point grids) and unstructured mesh models. Pillargrid models are used for fine-scale reservoir description and are designed to be suitable for reservoir flow simulation (Ponting, 1989). Unstructured meshes, especially tetrahedral models, are commonly used for geophysical modeling and migration on a large scale that includes both the reservoir and the overburden, and are designed to be suitable for fast simulation via ray tracing (Cerveny, 2001; Chapman, 2004). Unstructured meshes are also suitable for other modeling applications, e.g., geomechanical modeling, performed via finite-element simulation (Komatitsch et al., 2000; Dupros et al., 2010; Käser et al., 2010).

A pillargrid model is a logically-Cartesian tessellation of a Euclidean 3D volume in terms of hexahedral cells such that each cell can be identified by the integer coordinates (i, j, k). The (i, j) indices identify pillar curves, defined in Cartesian space, along which the hexahedral cell nodes lie. These pillar curves need not be linear. A geological surface is then the surface joining the nodal points at a common k value. Discontinuities in such a surface are supported by allowing adjacent cells that share a common pillar to have different nodal positions along the pillar curve, thus creating a step in the surface. Pillargrids are popular for reservoir simulation because the Cartesian cell ordering allows a simulator to efficiently address model cells, while preserving an ability to describe the geometrical and numerical discontinuities between cells necessary to represent fault networks and fine-scale stratification (Mamonov et al., 2007; Djikpesse et al., 2011).

The main limitation of pillargrids is that, while they are able to describe a broad range of reservoir geometries, there are many subsurface and reservoir configurations, such as salt domes, that pillargrids are topologically unable to describe. This limitation is compounded when a zone of interest is expanded to include a complex overburden in addition to the reservoir, with the result that most such models cannot be accurately described by a pillargrid. Hence pillargrids are typically not used for modeling applications that require overburden description, such as for seismic and geomechanical simulation. This is why, where particular geometry conditions exist, the constraint of being logically Cartesian is often dropped. This allows unstructured models to describe more accurately complex geometries.

The pillargrid and unstructured-mesh models are clearly quite different in nature and are used for different applications. However, modern seismic-to-simulation workflows contain iterative loops in which the workflow progresses from seismic to simulation and back again until the interpretation is complete. This type of progressive iteration necessitates conversions from detailed pillargrid models at the reservoir level to unstructured-mesh models that include both reservoir and overburden components.

There is little literature available on the complexities of pillargrid model conversion and potential algorithms to overcome them. In this article, we describe a method to convert pillargrid reservoir models into watertight unstructured models. In the next section, the conversion algorithm is detailed. Then, the proposed algorithm is demonstrated with a set of fine-scale field reservoir models and the performance of the algorithm is analyzed. Conclusions are drawn in the last section.

Method

Let us now describe the algorithm for converting a pillargrid model into an unstructured watertight model.

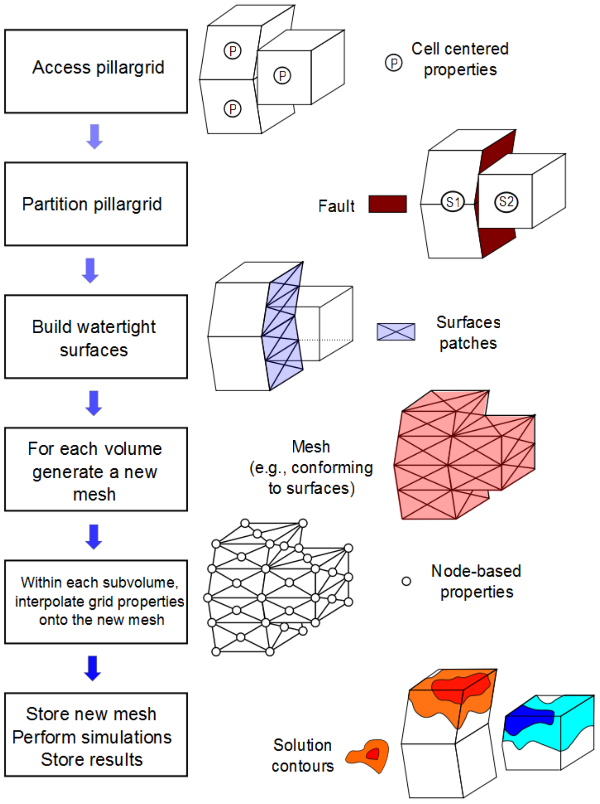
Access pillargrid
P
Cell centered properties
Partition pillargrid
Fault
S1
S2
Build watertight surfaces
Surfaces patches
For each volume generate a new mesh
Mesh (e.g., conforming to surfaces)
Within each subvolume, interpolate grid properties onto the new mesh
Node-based properties
Store new mesh Perform simulations Store results
Solution contours

Fig. 1: *The six phases of the pillargrid conversion algorithm. After accessing the pillargrid data, the pillargrid is partitioned into subvolumes. All the bounding surfaces of these subvolumes are triangulated using topological elements of the pillargrid. The subvolumes are then filled with points and properties.*

In the following we use a broader definition of unstructured mesh models, referred to as mesh models in this article, than is typically used in finite-element modeling. The model volume is partitioned into a non-intersecting set of subvolumes, each enclosed by a triangulated mesh surface, within which the property field is smoothly varying. Discontinuities in property fields and geometrical features can only occur across subvolume boundaries. Properties within each subvolume are described on a tetrahedral mesh, a Cartesian grid or, in principle, any geometry system capable of describing that property. Such property representations within a subvolume can overlap those of other subvolumes without ambiguity because the property description is only valid within the bounding surface of that subvolume.

A mesh model decouples property description from the geometry of subvolumes, allowing property population methods to be matched with subvolume boundary constraints. For example, properties can be represented by a regular grid even though the subvolume boundary is arbitrarily complex. This is possible because each bounding surface acts as “cookie cutters” that extracts only the part of the property grid that is interior to the bounding surface.

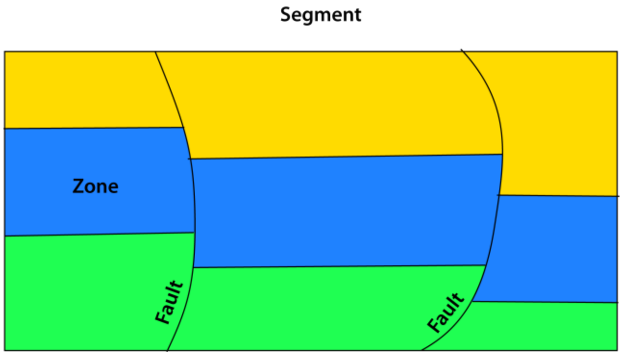

Fig. 2: *Original partition with three zones and three segments. Zones are horizontal volumes (yellow, blue and green) as segments are vertical volumes.*

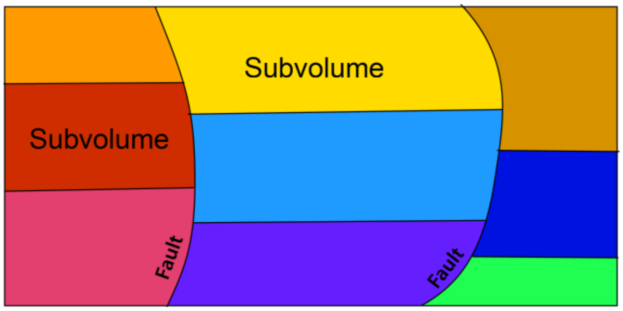

Fig. 3: *New partition with nine subvolumes.*

As shown in Figure 1, the conversion algorithm can be decomposed as follows:

- Partition the pillargrid into subvolumes
- Build the watertight surfaces
- Tessellate the volumes
- Fill the subvolumes with points and populating them with their properties

These conversion steps are described in more detail in the following subsections.

Partitioning the pillargrid - Figure 2 shows a description of the original partition of the pillargrid. It contains zones, faults and segments. A segment is a model subvolume that is bounded by fault pillars or volume boundaries. However, it can contain inner fault surfaces when the pillargrid contains partially penetrating faults. A zone is a horizontal volume containing a specific geological formation. We wish to partition a volume such that property continuity is ensured in each subvolume. This allows smooth property interpolation to be applied in the property population step. To achieve this, subvolumes are created as the intersection of zones and segments. The number of subvolumes is then equal to the number of segments multiplied by the number of zones (Figure 3).

Building the watertight surfaces - Watertight surfaces are triangulated surfaces in which every edge of each triangle is shared by at least one other triangle of the surface. Except for fault surfaces, all surfaces are built using a triangulation algorithm (Shewchuck, 1996). The fault surfaces are built based on the half-edge data structure described by Kettner (2009). Fault surface, external surface and horizon surface construction have a computational complexity on the order of $n^{2/3}$, where n is the number of cells in the pillargrid model, while undefined surfaces have a cost on the order of n.

Tessellating the volumes - Once the surfaces are built, it is possible to extract the boundary surface for each volume. The latter is a single closed watertight surface possibly made of several patch surfaces. The volume can then be tessellated with a tetrahedral mesh using the triangulated boundary as constraint, or can be filled by a regular grid that is trimmed by the bounding surface. Tessellation of the volumes into tetrahedra is achieved by a constrained Delaunay algorithm (Shewchuck, 1996).

Populating the properties - The properties for the pillargrid are defined on a cell-centered basis, i.e., the material properties of each cell are homogeneous throughout the cell. One way to define smooth properties throughout the volume is to interpolate from the pillargrid cell center values. We chose to use distance-weighted interpolation for general applications, though targeted interpolators may be warranted for particular applications. For efficiency, these cell-center values are inserted onto a Cartesian-grid data structure before interpolation in order to facilitate the rapid location of neighbors for any point in the volume. Typically, five points are used for interpolation.

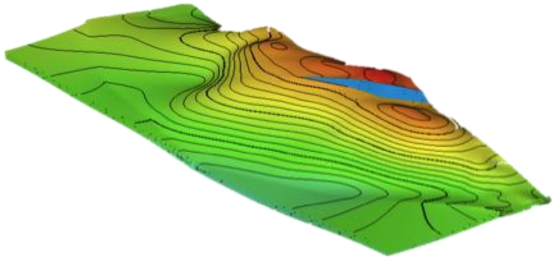

Fig. 4: Model B before conversion.

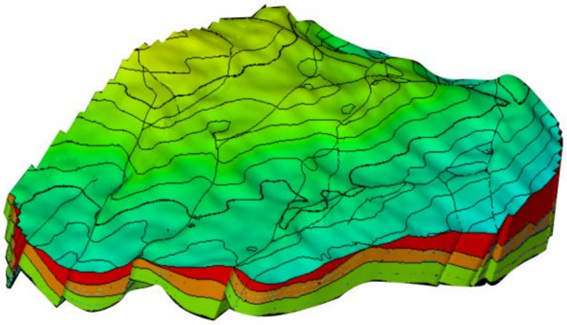

Fig. 5: Model E before conversion.

*Table 1: **Field** example characteristics.*

Models	# cells	# horizons	# faults	# volumes	Run times (s)
A	39733	25	0	24	240
B	60048	10	1	9	500.2
C	4158	23	27	1035	727
D	108800	35	11	340	2263.6
E	471240	7	23	120	2022.6

Field Results

After testing the method on synthetic examples (not presented here), we now present the conversion application to five field examples. The tests were performed on an Intel Xeon 3.20 GHz two-core processor with 12.0 GB RAM using Petrel 2010.1. Model characteristics are presented in Table 1. The five models are sorted by their geological complexity, which is not necessarily synonymous with a large number of cells (see, e.g., model C). Model A is a very basic model without any faults. Model B (Figure 4) has a large partially penetrating fault and some self-intersecting cells. Missing fault segments and a large number of faults made model C more difficult. Model D and model E (Figure 5) were selected because of the diversity of their difficulties: faults and partially penetrating faults, a large number of cells, and some collapsed cells. Figures 6 and 7 show the models B and E after conversion, respectively.

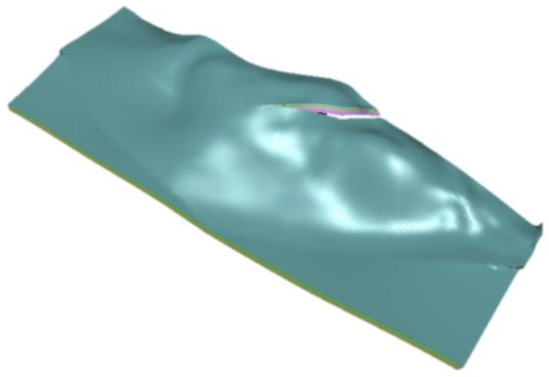

Fig. 6: Model B after conversion. Note the partially penetrating fault.

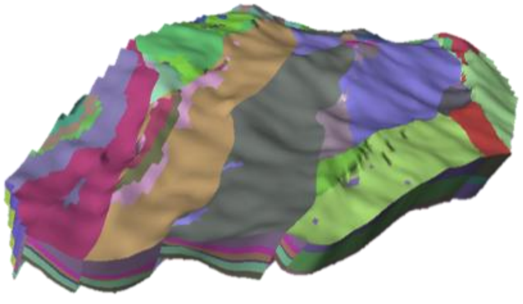

Fig. 7: Model E after conversion. Each color represents a different surface (a surface being defined by the two surrounding volumes). Note the presence of locally collapsed volumes.

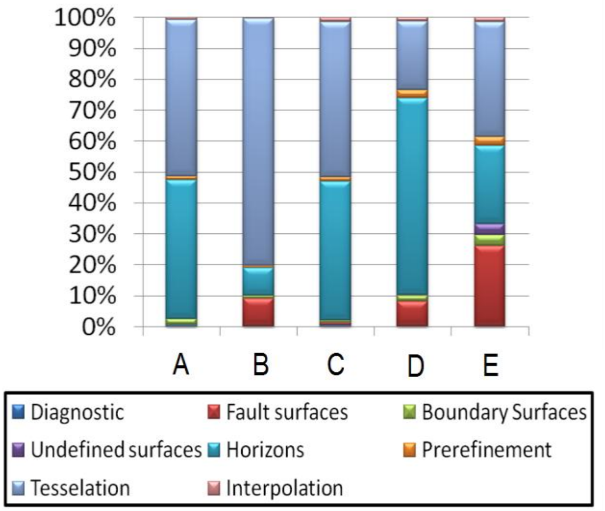

Fig. 8: Benchmarks for five field models. Global computing time are indicated in Table 1.

Figure 8 shows, for each of the five field reservoir models, the partition of the computation time over the different components of the conversion algorithm. Horizon triangulation consumes a significant fraction of the compute time. Three phases occupy most of the computing time: horizon construction, tessellation and fault surface construction. The computation time partition for model C is explained by the presence of many subvolumes, but few cells. Although fault surface construction possesses the highest computational complexity, but its fraction of the computation time is diminished by the low number of fault cells in comparison with the number of horizon cells. The diagnostic process (e.g., checking for self-intersecting cells) and the interpolation process do not have a significant impact on the overall computing time, as can be seen in these examples.

The impact of model size on computation time can be seen by comparing the run times of models D and E in Table 1. Model E has nearly five times more cells than model D, but has a slightly smaller run time. This is explained by the large number of horizons in model D — every layer was treated as an horizon when it was created — whereas there are only seven horizons for the sixty-three layers in model E. The considerable expense of horizon triangulation in model D dominates the larger cell count in model E.

Conclusions

We have presented how a pillargrid model can be successfully converted into a watertight model by using the original zone and segment structure of the pillargrid. We have also shown the results of such conversion on five field models with various geologic complexities. Such converted models are essential for performing, for instance, downhole finite-element simulations for reservoir-targeted seismic wave propagation, micro-seismic or electromagnetic wave modeling.

References

Cerveny, V., 2001. Seismic ray theory: Cambridge University Press, Cambridge, UK.

Chapman, C. H., 2004. Fundamentals of seismic wave propagation: Cambridge University Press, Cambridge, UK.

Djikpesse, H., Couet, B., and Wilkinson, D., 2011. A practical sequential lexicographic approach for derivative-free black-box constrained optimization: Engineering Optimization, 1029-0273.

Dupros, F., de Martin, F., Foerster, E., Komatitsch, D., and Roman, J., 2010. High-performance finite element simulations of seismic wave propagation in three-dimensional nonlinear inelastic geological media: Parallel Computing, 36, 308–325.

Käser, M., Pelties, C., Castro, C. E., Djikpesse, H., and Prange, M., 2010. Wavefield modeling in exploration seismology using the discontinuous galerkin finite-element method in HPC infrastructure: The Leading Edge, 29, 16–22.

Kettner, L., 2009. Halfedge data structures: CGAL References.

Komatitsch, D., Barnes, C., and Tromp, J., 2000. Wave propagation near a fluid-solid interface: Geophysics, 65, 623–631.

Mamonov, A. V., Couet, B., Bailey, W. J., Prange, M., Djikpesse, H. A., and Druskin, V., 2007. Optimal gridding: A fast proxy for large scale reservoir simulations: SPE/EAGE Reservoir Characterization and Simulation Conference.

Ponting, D. K., 1989. Corner point geometry in reservoir simulation: Joint Institute of Mathematics and its Applications/Society of Petroleum Engineers European Conference on the Mathematics of Oil Recovery, 45–65.

Shewchuk, J. R., May 1996. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator in Lin, M. C., and Manocha, D., Eds., Applied Computational Geometry: Towards Geometric Engineering:: Springer-Verlag, 203–222.