Common EM Model Format v0.1

March 8, 2017

Contents

1 About

The latest version of this specification can be retrieved from: `https://github.com/orgs/EMIWORG/ModelFormat`

1.1 Motivation

At the Third Magnetotelluric 3D Inversion workshop, which took place at the University of Aldo Moro in Bari, Italy, in mid-May, 2016, it was decided to form a small working group to explore the possibility of a common model format for EM modeling. This should facilitate exchange of models within community and ease their storage in data bases. It was decided that the binary model format should be based on HDF5 (hierarchical data format) (www.hdfgroup.org), because:

- it is an open, versatile, portable, scalable and widely used data model that can represent complex data and a wide variety of metadata;
- software libraries to read/write HDF5 files exist on a wide variety of platforms and can interact with many software languages (Fortran, C, C++, Python to name few);
- visualization packages such as Paraview and VisIt can render HDF5 files.

1.2 Reference

Please use following reference to cite this document:

Common Model Format for EM modeling and inversion, 2017, `https://github.com/EMIWORG/ModelFormat`

1.3 Contributions

The present document is a collaborative effort. Therefore, all discussions, suggestions and contribution are very welcome. These can be submitted via github link given above.

The following people (listed alphabetically) have contributed to the creation of this format: Alexander Grayver, Randall Mackie, Marion Miensopust, Federico Miorelli.

1.4 License

The format can be freely used for all purposes, including commercial, academic and governmental use. Accordingly, its usage implies no restrictions on the stored data. People who use the format are encouraged to reference its specification as stated in the Section 1.2.

Table 1: List of global attributes

Name	Mandatory	Type	Description
ModelName	Yes	string	Name of the model
MeshType	Yes	int32	1 - MESH_TYPE_RECTILINEAR; 2 - MESH_TYPE_TET; 3 - MESH_TYPE_HEX; 4 - MESH_TYPE_HEX_NON_CONFORMING see Table 2
TimeStamp	No	int64	POSIX time or epoch time. Defined as the number of seconds that have elapsed since 00:00:00 UTC, 1 January 1970.
Description	No	string	Description of the model

2 Specification

The model format consists of a number of mandatory and optional fields. Fields can be data sets or attributes and can be groups under common group name. The names, formats and units of mandatory fields and groups are defined below and cannot be changed. The names of optional fields must differ from the names of mandatory fields. Other than that, there are no restrictions for optional fields.

The format accommodates various mesh types as described in Section 2.1. Therefore, exact definition of the mandatory fields for a mesh depends on its type. Mandatory fields are not generally shared between different types, although fields have the same name. The only exceptions are fields listed in Section 2.1, which remain independent from the mesh type.

2.1 Common fields

2.1.1 Global attributes

Table 1 lists global attributes used to give the most generic description of the storeg model.

Table 2 lists mesh types, which the format accommodates. Note that in the current version only structured rectilinear meshes are fully specified. The specification for other types will follow in the next versions of the document.

2.1.2 Model reference

The model is anchored in Cartesian (projected) coordinates. The anchor point is taken as an origin of the model reference coordinate system.

- Northing, Easting, Altitude: Cartesian axes of the projected coordinate system. Altitude is in meters above sea level.

Table 2: Mesh types

Code	Description	Status
1	Structured rectilinear mesh	Supported
2	Unstructured tetrahedral mesh	Planned
3	Unstructured hexahedral mesh	Planned
4	Unstructured non-conforming hexahedral mesh	Planned

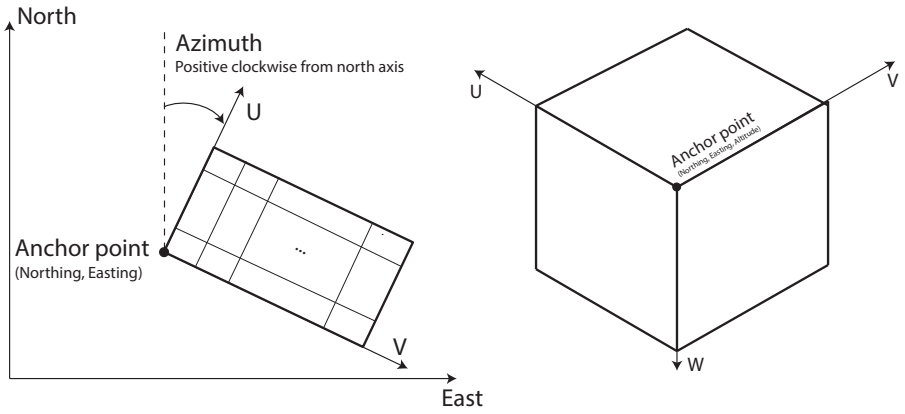

Figure 1: Coordinate system and notation adopted in the format.

- U, V, W: Cartesian axes of the model reference frame. U is positive up, V is positive right and W is positive down. A generic point in the model frame has therefore coordinates (u, v, w).

Figure 1 shows schematic illustration of the model coordinate system.

The coordinates of the anchor point and the projection type can be specified in the *Georeferencing* group defined in Table 3.

2.2 Dimensionality

The format is designed for most general three-dimensional case. Lower dimension models can of course be also stored by ignoring dimensions. For 2D, U

Table 3: Georeferencing group attributes

Name	Mandatory	Type	Description
AnchorNorthing	Yes	float64	Northing of the anchor point
AnchorEasting	Yes	float64	Easting of the anchor point
AnchorAltitude	Yes	float64	Altitude above sea level of the anchor point
Azimuth	Yes	float64	Azimuth of the model reference frame in degrees
Projection	No	string	Information about projection.

Table 4: Geometry group attributes

Name	Mandatory	Type	Description
NU	Yes	int32	Number of nodes in U direction
NV	Yes	int32	Number of nodes in V direction
NW	Yes	int32	Number of nodes in W direction

Table 5: Geometry group data sets

Name	Mandatory	Type	Shape	Description
NodesU	Yes	float64	NU	Coordinates of nodes in U direction
NodesV	Yes	float64	NV	Coordinates of nodes in V direction
NodesW	Yes	float64	NW	Coordinates of nodes in W direction

is assumed to be the strike direction. Accordingly, for 1D both U and V are ignored.

3 Structured rectilinear grid

A mesh is defined by specifying its geometry and physical properties for each cell. This is done via *Geometry* and *Properties* groups as described below.

3.1 Geometry group

The geometry of a rectilinear mesh is uniquely defined by three data sets *NodesU*, *NodesV* and *NodesW* specifying coordinates of the cell nodes in each direction relative to the anchor point. The sorting of all the arrays is consistent with the orientation of the U, V, W axes, including the storage order of the properties. Group description is given in Tables 4 and 5.

3.2 Properties group

Each can be assigned a number of properties. The mandatory properties include a cell type and at least one resistivity value. The group attributes and data sets are given in Tables 6 and 7.

Cell type property can be used to distinguish air, sea and subsurface or to specify active/inactive inversion domains, etc.

Cells are counted with respect to U, V and W. For instance, I,J,K=1,1,1 corresponds to the anchor point in the lower-left corner of the model, at the model top point. In this point U=0,V=0,W=0. Indices increase accordingly with U,V,W.

For isotropic models, the resistivity will be specified using the property name of *Rho*. The resistivity for transversely isotropic model will be specified using the property names of *RhoH* and *RhoV*, while the resistivity for a diagonally

Table 6: Properties group data sets

Name	Mandatory	Type	Shape	Description
CellType	Yes	int64	[NU-1, NV-1, NW-1]	Cell type property
Rho	No	float64	[NU-1, NV-1, NW-1]	Isotropic resistivity
RhoV RhoH	No	float64	[NU-1, NV-1, NW-1]	Transversely isotropic resistivity
RhoU RhoV RhoW	No	float64	[NU-1, NV-1, NW-1]	Anisotropic resistivity
Alpha Beta Gamma	No	float64	[NU-1, NV-1, NW-1]	Euler angles for general anisotropic resistivity

Table 7: Property dataset attributes

Name	Mandatory	Type	Description
Unit	Yes	string	Physical unit of the property
BlankValue	No	float64	Blank value identifier. Default is NaN.

anisotropic model will be specified using *RhoU*, *RhoV*, and *RhoW*. Model readers should first query the file for *Rho*, then proceeding to the other levels of anisotropy in ascending order. For the most general case, Euler angles can also be specified.

3.3 Examples

In the directory *./examples/rectilinear/* you will find as an example a MATLAB script which writes COMMEMI3D-2 model into HDF5 file with the specified format.

4 Visualization

Popular visuzalization software such as Paraview and VisIt support HDF5 format via eXtended Data Model Format (XDMF: www.xdmf.org). XDMF uses XML to describe data stored in a HDF5 file. Once XDMF is loaded, the program will refer to the specified HDF5 data sets and read relevant data from them. Due to its simple and intuitive structure, XDMF can be easily generated along with the HDF5 file. An example of an XDMF file that describes COMMEMI3D-2 model stored in the *commemi.h5* file and resulting grid rendered by the Paraview (Figure 2) can be found in the directory *./examples/rectilinear/*.

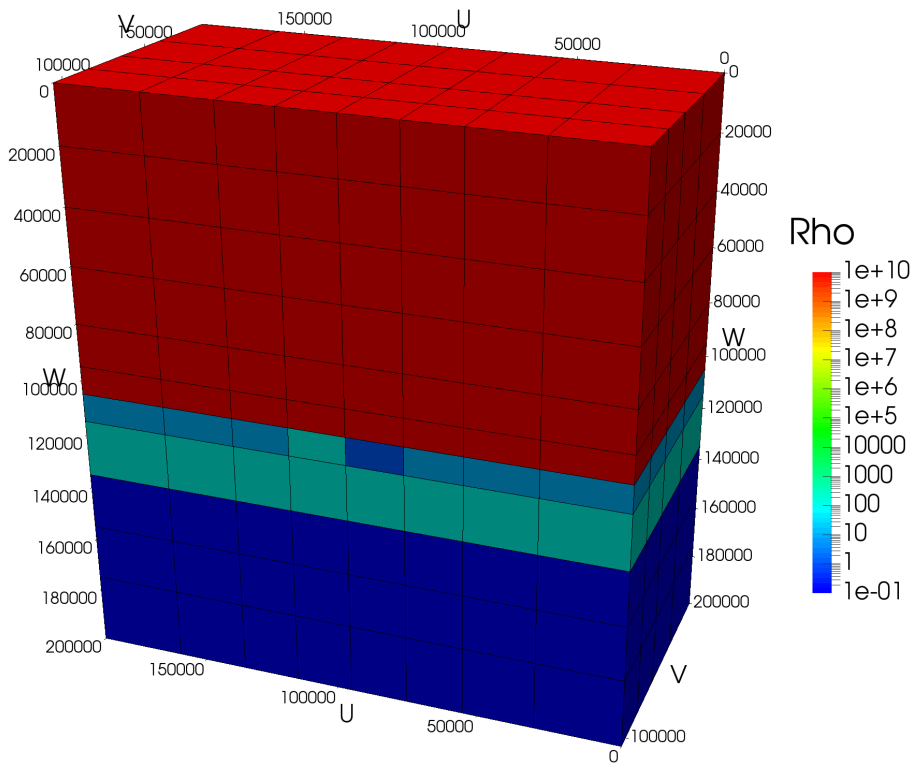

Figure 2: Cut through the COMMEMI3D-2 model loaded in Paraview with XDMF and HDF5.