Archiving GIS-based Vectors Formats in PDS4 using GeoCSV

revision 4, July 2018

Trent Hare, thare@usgs.gov

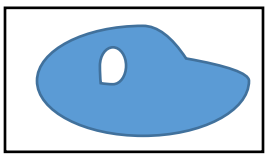
Introduction to GIS vector formats

The PDS has long had a need to support Geographic Information System (GIS) vector (point, line, and polygon) style of format. Currently, missions and funded projects are forced to archive such data types within "miscellaneous" or "extras" archival directories. Many of these files are not discoverable or contain the appropriate metadata as provided by a proper NASA Planetary Data System (PDS v4) label (see p. 12 of the Data Providers Handbook [1]). Data which have been unofficially archived in this manner include image footprints, Lunar mare boundaries, Titan channels, Apollo 17 traverse lines, and layers which comprise of a geologic map (linear structure and geologic polygonal boundary units). Here we introduce a method which builds upon PDS4's approved variable width ASCII table support to allow archival for GIS vector files (here, informally called GeoCSV [2]).

To reiterate, this whitepaper is only addressing GIS vector formats which define points, lines, or polygons as a series of Longitude and Latitude vertices (or points). PDS4 allowable tables which have a Longitude and Latitude field can easily support point features (e.g. crater locations) but these tables are currently insufficient for lines and polygons. Alternative concepts proposed that a line or polygon file could be a PDS4 allowable table with multiple separate but linked Lon/Lat vertex tables per feature. With potentially thousands of features per layer, it would require this master PDS4 table to hold the feature's attributes (type, length, width, area, etc.) and then many thousands separate tables to list the vertices. This quickly becomes unreasonable to support. More importantly, these lines and polygons also need to support multiple parts for a single feature (see Figures 1 and 2) which a single table of vertices does not easily allow. Thus, to support complicated multipart geometric features within a PDS4 variable length ASCII table, we will include a single field which uses the established Well-Known Text (WKT) geometry string.

Figure 1. Shows that lines can have multiple parts. This multipart line is really one feature. For an example vertex listing, see Figure 4.

Figure 2. Shows that polygons require understanding of being closed and show needed support for holes (negative space). This example is one feature. For an example vertex listing, see Figure 4.



Well-Known Text geometries: The Well-known Text (WKT) geometry standard is a text markup language for representing vector geometries and this standard is used broadly across GIS libraries and applications. Defined within the International Organization for Standardization (ISO) standard 19125, a WKT geometry specifies a common storage and access model of mostly two-dimensional (2D) and

optionally three-dimensional (3D) geometries including: point, line, polygon (Figure 3) and multi-point, multi-line, multi-polygon (Figure 4).

Coordinates used within a geometry string may be 2D (Lon, Lat or X, Y) or 3D (Lon, Lat, Z or X, Y, Z). The order of coordinates in WKT, as shown above, is strictly enforced. While map projected Cartesian coordinates in meters are supported (e.g. X, Y, Z), for PDS4 archives it is recommended WKT geometries are listed in decimal degrees (Longitude, Latitude). The precision for all values will be determined by the data provider. WKT strings can easily support 64-bit precision (~16 decimal digits), but in general, 32-bit precision (~7 decimal digits) is recommended. The total WKT string length or the number of features in one PDS table will not be limited, however, it is recommended to remove unnecessary details for the archive when possible. Geometries which are empty and contain no coordinates can be specified by using *EMPTY* after the type name e.g., "LINESTRING EMPTY". Lastly, it is recognized that this method to encode geometries is not optimized for direct use. Thus, users are encouraged to convert this archival format to a more performant vector format for general use.

Geometry primitives (2D)

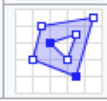
Type	Examples	
Point		`POINT (30 10)`
LineString		`LINESTRING (30 10, 10 30, 40 40)`
Polygon		`POLYGON ((30 10, 40 40, 20 40, 10 20, 30 10))`
		`POLYGON ((35 10, 45 45, 15 40, 10 20, 35 10), (20 30, 35 35, 30 20, 20 30))`

Figure 3. Shows **single-part** WKT geometries. Order for WKT string will always be Longitude (X) followed by Latitude (Y).

Image credit: https://en.wikipedia.org/wiki/Well-known_text

Multipart geometries (2D)

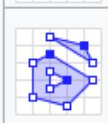
Type	Examples	
MultiPoint		`MULTIPOINT ((10 40), (40 30), (20 20), (30 10))`
		`MULTIPOINT (10 40, 40 30, 20 20, 30 10)`
MultiLineString		`MULTILINESTRING ((10 10, 20 20, 10 40), (40 40, 30 30, 40 20, 30 10))`
MultiPolygon		`MULTIPOLYGON (((30 20, 45 40, 10 40, 30 20)), ((15 5, 40 10, 10 20, 5 10, 15 5)))`
		`MULTIPOLYGON (((40 40, 20 45, 45 30, 40 40)), ((20 35, 10 30, 10 10, 30 5, 45 20, 20 35), (30 20, 20 15, 20 25, 30 20)))`

Figure 4. Shows supported **multi-part** WKT geometries. Order for WKT string will always be Longitude (X) followed by Latitude (Y).

Image credit: https://en.wikipedia.org/wiki/Well-known_text

Recommended methods to convert GIS formats to PDS4 GeoCSV

Nearly all GIS applications support conversion of geometries to the WKT string representation. But for PDS4 archives, we will support a freely available stand-alone application to assist users.

1. The conversion from dozens of common GIS formats to this geometry-capable PDS4 table can be accomplished using the open source library Geospatial Data Abstraction Library (GDAL [3]). The routine, *ogr2ogr*, works on all major platforms (Windows, Macintosh, and Linux). Field description and field types will be automatically defined within the created PDS4 label. For PDS4 label entries that cannot be automated by the GDAL driver (e.g. mission name), a user provided PDS4 XML template file, with additional metadata, can be used during conversion. This allows the data provider to use existing PDS4 tools like the PDS Label Assistant for Interactive Design (PLAID) and On-Line Archiving Facility (OLAF) to create the initial PDS4 template. Also, user-defined variables within the provided template label are supported and can be set during conversion. Lastly, the target body and radius, and when applicable, the map projection (e.g. equirectangular), will be written to the Cartography discipline dictionary (CART) section of the PDS4 label.

   An example conversion from a GIS Shapefile to a PDS4 GeoCSV (variable-width table):

   ```
   $ogr2ogr -f PDS4 out_PDS4.xml -co TEMPLATE=input_pds4_template.xml
   in_shapefile.shp -lco GEOMETRY=AS_WKT
   ```

   An example showing the reverse, PDS4 GeoCSV (table) conversion back to the Esri Shapefile format:

   ```
   $ogr2ogr -f "Esri Shapefile" out.shp in_PDS4.xml -oo WKT=wkt_field
   ```

   An example PDS4 table conversion to the GeoJSON format using existing Longitude and Latitude fields as no WKT string available in the PDS4 table:

   ```
   $ogr2ogr -f "GeoJSON" out.json in_PDS4.xml -oo LAT=lati -oo LON=long
   ```

   For more examples please see [3, 4] and for an introductory presentation see [5].

2. Because the format is a simple table, creating custom software to read and write this PDS4 table should also be easily accomplished.

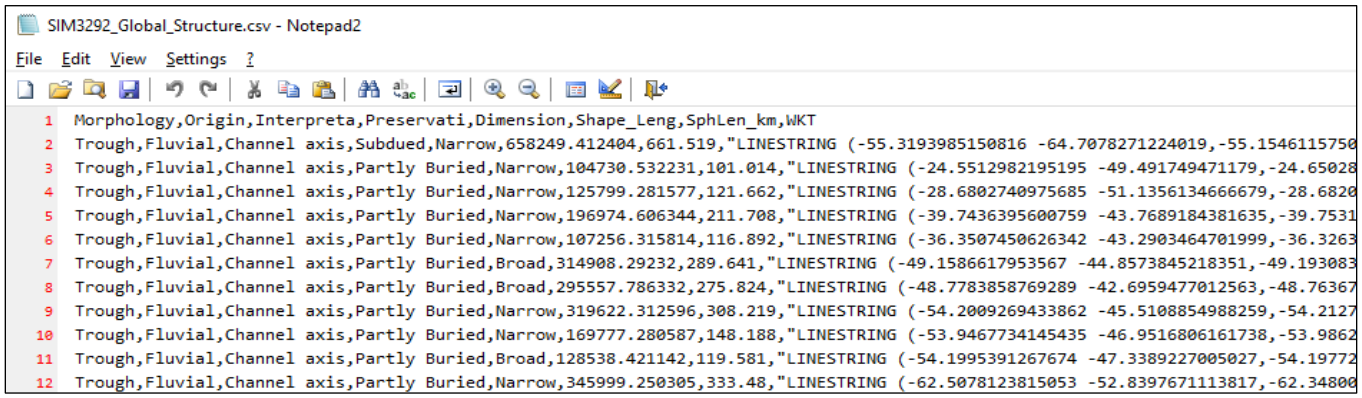


Figure 5. Shows what a final table will look like. Here is shown a geologic structural line file. The line's attributes are listed as simple table fields separated by a comma delimiter. The last field is reserved for the WKT string (truncated due to length). A typical PDS4 header (XML formatted) will also be required to define the tables structure (field names, description, and field types).

References:

[1] The PDS4 Data Provider's Handbook, Guide to Archiving Planetary Data Using the PDS4 Standard, 2020, Version 1.14.0 URL: https://pds.nasa.gov/datastandards/documents/dph/.

[2] Gaddis, L. R., & Hare, T. M. (2019). *Archiving GIS-type products in the PDS.* Paper presented at the 4th Planetary Data Workshop, Flagstaff, AZ, URL: https://www.hou.usra.edu/meetings/planetdata2019/pdf/7045.pdf

[3] PDS4 – Nasa Planetary Data System (Version 4), GDAL documentation, 2020, URL: https://gdal.org/drivers/raster/pds4.html

[4] Hare, Trent, 2019, Vector GIS introduction for PDS4 conversion using GDAL, GitHub Wiki, URL: https://bit.ly/PDS4_Vectors.

[5] Hare, Trent and Gaddis, Lisa, 2018, GIS Data in PDS4, PDS Council Meeting, URL: http://bit.ly/GIS4PDS4.