GDAL Tips and Tricks
Jeremy Mayeres - FOSDEM 2018
Overview

• Introduction
• Installation
• Using with Python
• Misc Tips
• Cloud-optimized GeoTIFF and VSICurl
• Where to go for help
Introduction
About me

- Team lead for Data Pipeline in Berlin at Planet Labs Germany
- Living in Berlin since 2014, working for BlackBridge/Planet
- Belgian-born
- Raised in Florida
- Studied Computer Science in Orlando at UCF (2008-2012)
- Attended FOSDEM 2015, 2016, 2017
- First time speaking at conference
About GDAL
Geospatial Data Abstraction Library

- Initially created by Frank Warmerdam (my colleague in SF)
- Now managed by OSGeo (Open Source Geospatial Foundation)
- C/C++ library
- Reading/writing vector (e.g. shape) and raster (e.g. image) data
- Supports >150 raster formats, >90 vector formats
- Used by most GIS software (incl. ArcGIS, GRASS, QGIS, SNAP…)

GDAL
Common GDAL Utilities

- `gdalwarp`
  - Used for mosaicing, reprojection and warping
- `gdal_translate`
  - Change from one raster format to another
- `ogr2ogr`
  - Change from one vector format to another
- `gdalinfo`
  - Get information about a file (including statistics)
Installation
Installation
MacOS

• Install Homebrew (if not already installed from brew.sh)
  brew tap osgeo/osgeo4mac
  brew install gdal2 gdal2-python

Contribute:
https://github.com/OSGeo/homebrew-osgeo4mac
Installation
Ubuntu

• Use the UbuntuGIS PPA
  
  sudo add-apt-repository ppa:ubuntugis/ppa
  sudo apt-get update
  sudo apt-get install gdal-bin

• Alternatively, use Anaconda:
  
  conda install -c conda-forge gdal
Installation
Docker or Vagrant

Why?
• Using VM/Container isolates system dependencies
• Upgrading certain packages can break things
• Repeatability to get a working config
• Avoid having to use Python virtualenv

How?
• As simple as making instructions in previous slide a script in the Dockerfile or Vagrantfile
Python and GDAL

- Since GDAL 2, powerful bindings for GDAL Translate, Warp, etc.
  - No need for command line call!
  - Takes arguments like the command line call
    - `gdal.Warp(dst_ds, src_ds, options=warp_options)`
- Numpy integration
  - `band.ReadAsArray(xoff, yoff, xcount, ycount)`
  - Gives you a Numpy array
- `ogr2ogr` is `gdal.VectorTranslate`
- See [GDAL RFC 59.1](https://gdal.org/)
Or don't use GDAL directly

- For raster data handling: **rasterio**
  - Pythonic
  - GDAL under the hood
  - Easier to use

- For vector/shape handling: **Shapely** and **Fiona**
  - Pythonic
  - Shapely: GEOS dependency, Fiona: GDAL/OGR
  - Easier to use
  - Load GeoJSON, WKT objects and easily perform operations like validation, intersection/union, area calculation, transform...
Misc Tips
Tips

• Read the code of GDAL's utilities and Python scripts!
  ○ Great way to learn how to use GDAL's API
• Buffer geometries by zero to fix geometry issues
  ○ `valid_geom = invalid_geom.Buffer(0)`
• Use VRTs (Virtual Datasets) to treat imagery tiles as one mosaic
  ○ Break up larger imagery into smaller tiles
  ○ Work with them all together using one VRT
  ○ XML that describes the operations
  ○ Needs common projection!
Cloud-Optimized GeoTIFF
Cloud-Optimized GeoTIFF (COG)

- GeoTIFFs for better cloud processing!
- Regular GeoTIFF, leveraging existing features
  - Tiling
  - Overviews
- Optimized for using HTTP GET range requests
- Allows for streaming large rasters, downloading only needed data
- Used with GDAL's VSICurl driver
- OpenAerialMap, DigitalGlobe, and Planet provide imagery as COG
- Read more at [http://www.cogeo.org/](http://www.cogeo.org/)
How to make a Cloud-Optimized GeoTIFF

• Create internal overviews (if not already created):
  `gdaladdo -r average in.tif 2 4 8 16`

• Create Cloud-Optimized GeoTIFF
  `gdal_translate in.tif out.tif`
  `-co TILED=YES`
  `-co COPY_SRC_OVERVIEWS=YES`
  `-co COMPRESS=LZW`

• Host it on HTTP 1.1 server supporting Range Requests (Byte Serving)
  ○ Amazon S3, Google Cloud Storage, and Azure support this
How to use a Cloud-Optimized GeoTIFF

• If you're already using VSICurl and using a window, nothing is different

• Examples:
  ○ https://trac.osgeo.org/gdal/wiki/CloudOptimizedGeoTIFF#Readingablockofpixelsatfullresolution
  ○ https://gist.github.com/sgillies/7e5cd548110a5b4d45ac1a1d93cb17a3
Using VSICurl

- Instead of file path, add /vsicurl/ in front of the URL
- Example: gdalinfo
  
  /vsicurl/http://landsat-pds.s3.amazonaws.com/L8/001/003/LC80010032014272LGN00/LC80010032014272LGN00_B1.TIF
Where to get help
Where to go for help

- https://gis.stackexchange.com/
- #gdal IRC channel on irc.freenode.net
- gdal-dev@lists.osgeo.org
- gdal.org

Useful links:
- GDAL Intro series by Rob Simmon
- Python GDAL/OGR Cookbook
- Awesome Geospatial List (shameless self-promotion)
Acknowledgements

- Frank Warmerdam
- Even Rouault
- Christian Wygoda
- Chris Holmes
- Robert Simmon
- Kevin Wurster
- SW Engineering teams at Planet
- Icons CC 3.0 BY Dave Gandy - Font Awesome
- Imagery from Planet
Thank You

Find me:

@Jerr
jerr0328
https://jeremy.berlin
jeremy@planet.com

Cat: