The benefits of COG* outside the cloud

*Cloud Optimized GeoTIFF

25th August 2022 - Firenze

FOSS4G

FIRENZE 2022

About me...

© OpenStreetMap contributors

Nicolas Rochard - Geo2France

- Collaborative opendata platform in Hauts-de-France
- SDI* powered by GeOrchestra
- In charge of land cover - land use data production and analysis

#QGIS #GRASS #GDAL #PostGreSQL #PostGIS #QGIS Server #MapLibreGL

* *Spatial Data Infrastructure*

Use Cases

Pandemic context :

- Maintain the efficiency of the GIS department
- Load quickly a large amount of raster data by VPN with bandwidth issue and random speed...

Streamline disk usage :

- Raster tiles pre-calculated for GeoServer (SDI)
- Raster mosaic files with overview for GIS Desktop Software

OGC webservices & file format comparison

WMS-WMTS

- ✅ Displaying fast (750ms to 1250ms)
- ❌ No band order control
- ❌ No stretching option

WCS

- ✅ Raw data
- ✅ Band order control
- ✅ Stretching option
- ❌ Sloooowwwwwww (2500ms to 4500ms)

JP2 / GTIFF

- ✅ Band order control
- ✅ Stretching option
- ❌ GTIFF with overview can be fast displayed at small scale but first loading can take a while
- ❌ JP2 unusable without proprietary drivers

What about COG ?

A Cloud Optimized GeoTIFF (COG) is a regular GeoTIFF file, aimed at being hosted on a HTTP file server, with an internal organization that enables more efficient workflows on the cloud. It does this by leveraging the ability of clients issuing HTTP GET range requests to ask for just the parts of a file they need.

source : www.cogeo.org

➡ **Streaming : you don’t have to download all the data before vizualise them**

COG without Cloud ? How ?

- *Lockdown context* : COG was processed on a remote computing server without any GUI.
- Need to check visually if COG is not corrupt and compression option are right… Simply drag n drop into QGIS from remote network drive and …

No S3 bucket

No HTTP Server

Wait for it !

Let’s continue without Cloud

Loading COG in different applications :

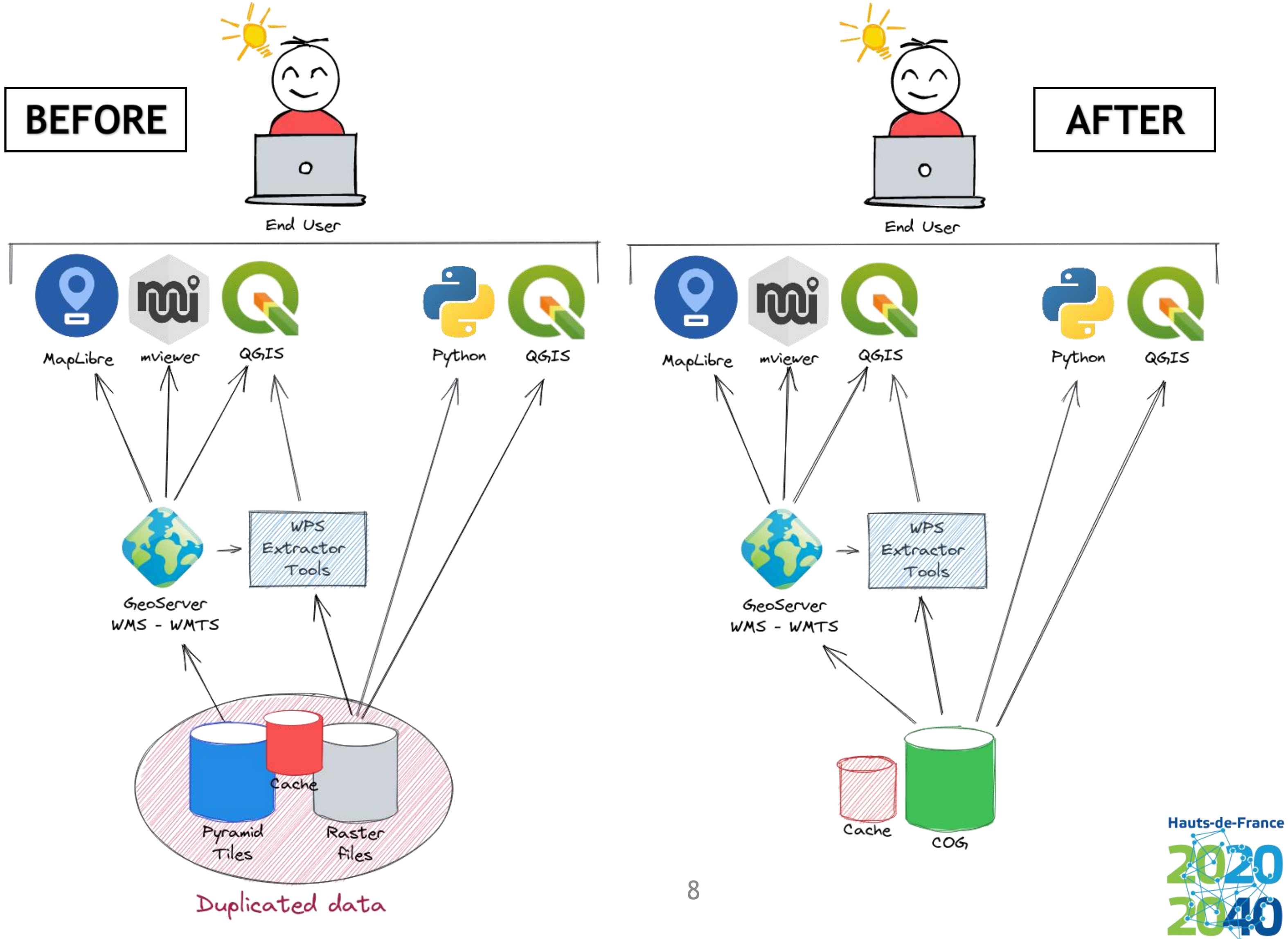
- *QGIS Server : no cache - WMS and WMTS fast display time*
- *GeoServer* : as fast as image pyramid or from tiles cached*

→ **No need to add new instance such as HTTP Server or S3 storage at the moment on our SDI.**

- *ArcGIS Desktop : it works and very efficient too but not as fast as QGIS Desktop*

** As GeoTIFF Raster Data Sources in a GeoServer Store*

COG optimize SDI architecture



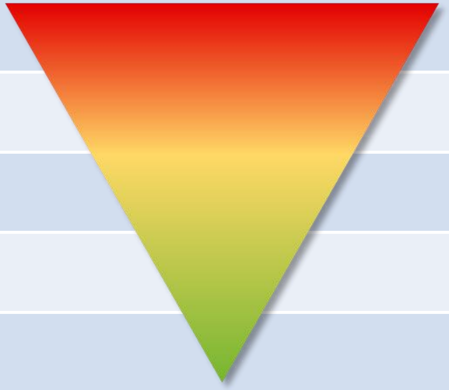
COG compression benchmark (8 bits raster imagery)

Compression Method	Quality	Size	Computing Time
N/A	N/A	159.0 Go	N/A
LZW		74.7 Go	81 min
DEFLATE		60.6 Go	128 min
JPG	75	3.7 Go	84 min
JPG	90	6.9 Go	84 min
WEBP	*75*	*2.5 Go*	*170 min*
WEBP	*90*	*6.1 Go*	*170 min*
JP2	*N/A*	*3.9 Go*	*N/A*

🟨	Lossy compression	🟪	Both : Lossy & Lossless compression
🟦	Lossless compression		

COG compression benchmark (8 bits raster imagery)

Depending compression method, reading performance varies :

Compression Method	Display Time
No compression	
WEBP	
LZW	
DEFLATE	
JPG	

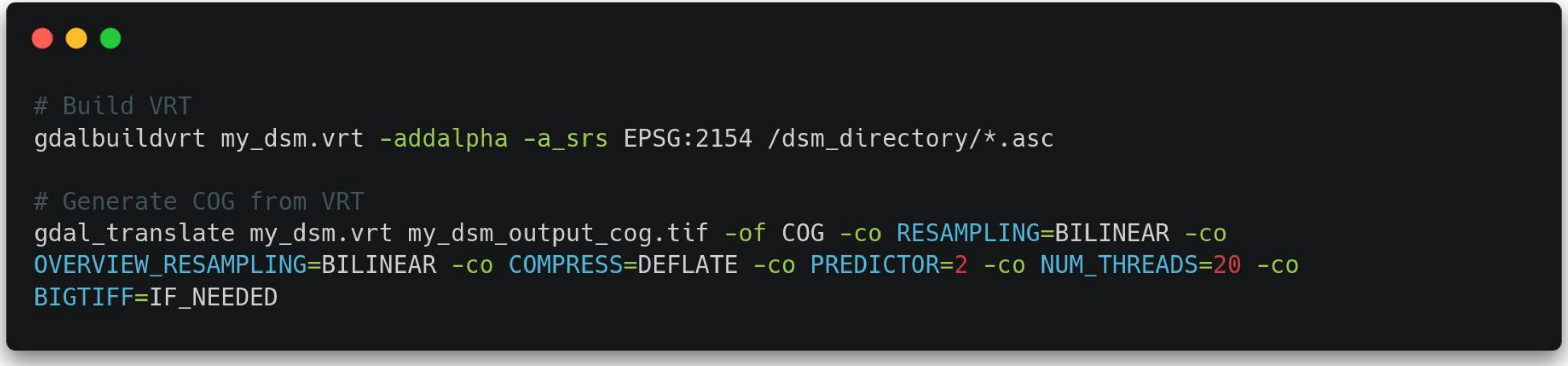
GDAL Cheat Sheet

Don't be afraid by command lines

read the docs : https://gdal.org/drivers/raster/cog.html

1 BAND RASTER

DSM/DEM : Digital Surf/Elev Model

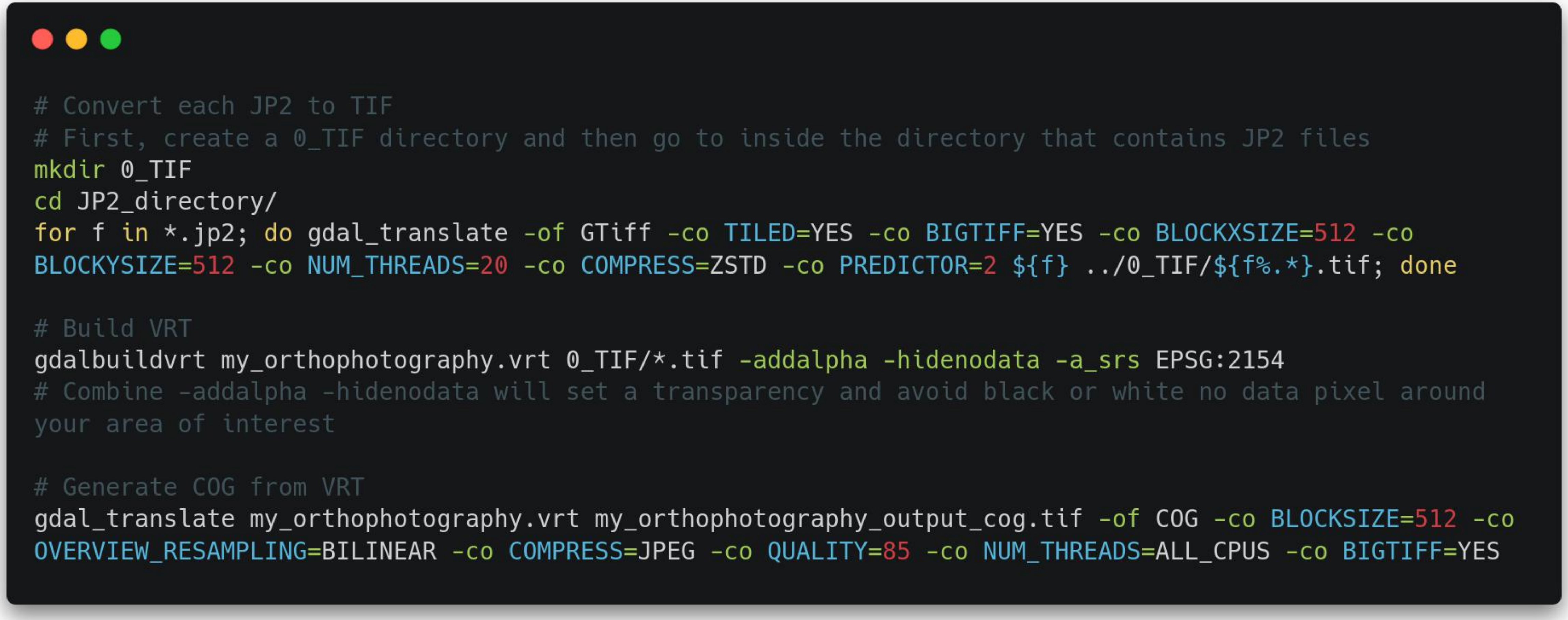


```
# Build VRT
gdalbuildvrt my_dsm.vrt -addalpha -a_srs EPSG:2154 /dsm_directory/*.asc

# Generate COG from VRT
gdal_translate my_dsm.vrt my_dsm_output_cog.tif -of COG -co RESAMPLING=BILINEAR -co
OVERVIEW_RESAMPLING=BILINEAR -co COMPRESS=DEFLATE -co PREDICTOR=2 -co NUM_THREADS=20 -co
BIGTIFF=IF_NEEDED
```

Orthoimagery / VHR Imagery

3 BANDS RASTER



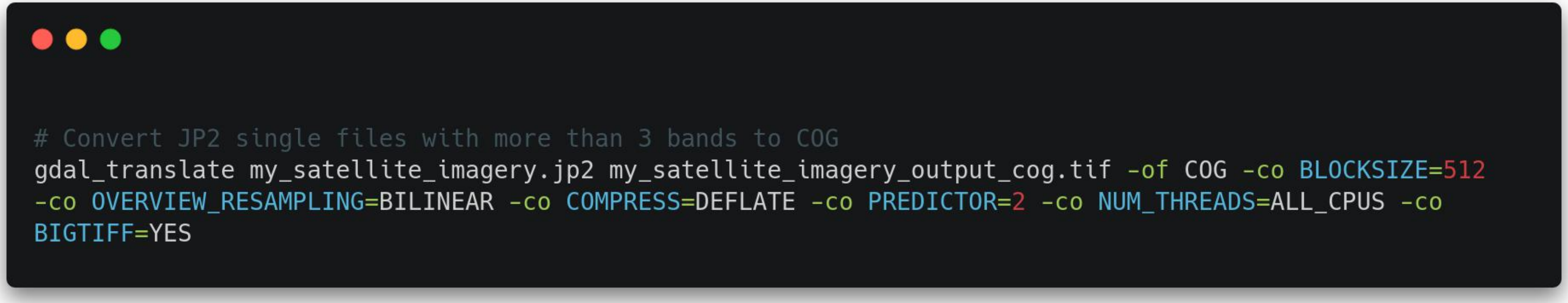
```
# Convert each JP2 to TIF
# First, create a 0_TIF directory and then go to inside the directory that contains JP2 files
mkdir 0_TIF
cd JP2_directory/
for f in *.jp2; do gdal_translate -of GTiff -co TILED=YES -co BIGTIFF=YES -co BLOCKXSIZE=512 -co
BLOCKYSIZE=512 -co NUM_THREADS=20 -co COMPRESS=ZSTD -co PREDICTOR=2 ${f} ../0_TIF/${f%.*}.tif; done

# Build VRT
gdalbuildvrt my_orthophotography.vrt 0_TIF/*.tif -addalpha -hidenodata -a_srs EPSG:2154
# Combine -addalpha -hidenodata will set a transparency and avoid black or white no data pixel around
your area of interest

# Generate COG from VRT
gdal_translate my_orthophotography.vrt my_orthophotography_output_cog.tif -of COG -co BLOCKSIZE=512 -co
OVERVIEW_RESAMPLING=BILINEAR -co COMPRESS=JPEG -co QUALITY=85 -co NUM_THREADS=ALL_CPUS -co BIGTIFF=YES
```

Satellite Imagery

4+ BANDS RASTER



```
# Convert JP2 single files with more than 3 bands to COG
gdal_translate my_satellite_imagery.jp2 my_satellite_imagery_output_cog.tif -of COG -co BLOCKSIZE=512
-co OVERVIEW_RESAMPLING=BILINEAR -co COMPRESS=DEFLATE -co PREDICTOR=2 -co NUM_THREADS=ALL_CPUS -co
BIGTIFF=YES
```

- JPG compression cannot be used as there is more than 3 bands
- Use DEFLATE (safer option)
- Could use ZSTD if your GIS application or Python libraries are uptodate

TIPS & TRICKS

Good Pratices

- **GDALBUILDVRT** and use **GDAL_TRANSLATE** is more efficient than use directly **GDALWARP**
- As JP2 (or ECW) is already compressed, to avoid image degradation, set compression level quite low 85~90
- If you start from native raw TIF, then adjust around 75-80 compression **QUALITY**
- If **ZSTD** is not working from JP2 to TIF, then use **DEFLATE**
- **RESAMPLING** method depending of user choice but BILINEAR offer beautiful rendering. Some user may prefer AVERAGE.

Conclusion

COG offer an opensource clever format suitable for many geospatial applications even outside of a cloud context.

- *Amazing performance on Samba / CIFS / NFS without any SDI or OGC map server*
- *Improve display time but also processing time for image classification or analysis*
- *Versatile for a wide range of raster data sources : all our data are now converted to COG*
- *COG files are bigger than JP2 or ECW files*
- *May use more CPU on server side when using with QGIS Server / GeoServer (but less disk space as cache is not necessary)*
- *Be careful about compression method depending of degree of compatibility expected. Reason why ZSTD or JPEG-XL are not retained in these use cases.*

Hauts-de-France 2020 2040

Région Hauts-de-France

Géo2France

Thank you for your attention 🙏

✉ nicolas.rochard@hautsdefrance.fr

@DoctorWho80

www.linkedin.com/in/nicolas-rochard

github.com/Doctor-Who/cog-recipes

FOSS4G FIRENZE 2022