君は新しいGDALを知っているか

自己紹介

- ありた （a.k.a. tohka）
- Twitter（現X）：@tohka383

GDAL/OGR

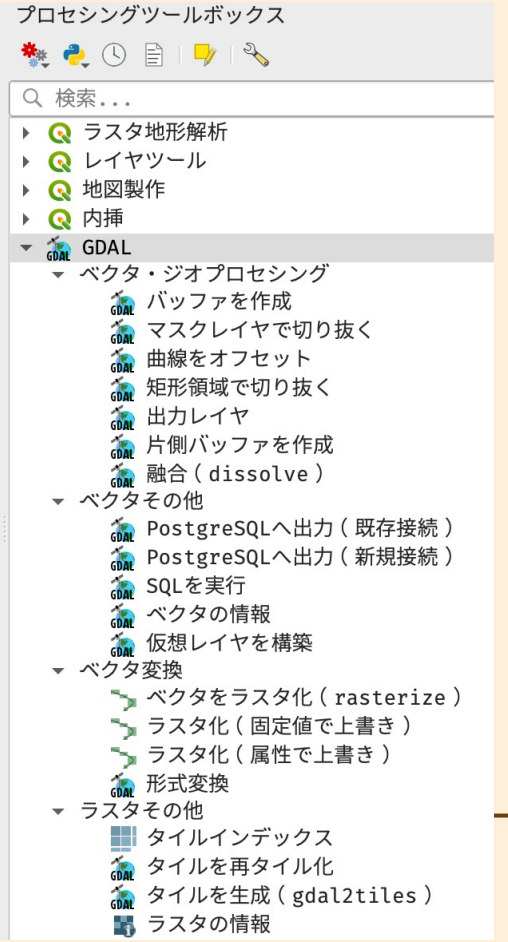
- 1998年に開発がスタート
- ラスタデータやベクタデータの変換などが可能なツール
  - ファイルの形式変換、座標参照系の変換（再投影）
  - 点データからラスタ化、ラスタデータからポリゴン化
  - ラスタ値のスケーリング、再分類 (reclassify)
  - nodata 領域を周囲セル値から補間
  - 陰影起伏図、傾斜量図、等高線の作成
  - ベクタデータの条件フィルタリング、 SQL 文の実行
- 155種のラスタ形式、83種のベクタ形式（ドライバ数）に対応
- 基本はコマンドラインツール群
  - C/C++/Python をはじめ多種言語にバインディング

GDAL/OGR の様々なコマンド

- ラスタに関するコマンド（一例）
  - `gdalinfo` ラスタデータの情報を確認
  - `gdal_translate` 異なるフォーマットに変換
  - `gdalwarp` データの再投影
  - `gdal_grid` 散布図状の点データからラスタ作成
  - `gdaldem` DEM から陰影起伏図や傾斜量図など作成
  - `gdal2tiles` XYZ / TMS ラスタタイルを作成
- ベクタに関するコマンド（一例）
  - `ogrinfo` ベクタデータの情報を確認
  - `ogr2ogr` フォーマット変換、再投影等

GIS ツールの縁の下の力持ち

- FOSS4G ツールや商用ソフトにも使われている
- QGIS にも使われている
  - プロセシングツールの GDAL ツリー下
  - ダイアログ下部に実行コマンドが表示
  - 詳細パラメータの「追加のコマンドライン パラメータ」に当該コマンドのオプションを追加することが可能
  - 下図は範囲を (0, 0) - (100, 100) とし 解像度 5m/px と指定した例（青ハッチ部）



GDAL/OGR コンソールコール

```
gdal_grid -l layer_name -a linear:radius=-1.0:nodata=0.0 -ot Float32 -of GTiff
-txe 0 100 -tye 0 100 -tr 5 5 path_to_data_file /private/var/folders/l6/
r4dhxxf971qfr9swjfzy2ssm0000gn/T/processing_EBsCwH/
f8a9337c3d354ac4af9183849c580a24/OUTPUT.tif
```

コマンドを使うには

- インストール方法にはいくつか種類があるが、QGIS をインストールすると GDAL/OGR も一緒にインストールされるので、それを使うと楽

- Windows （ Standalone インストーラの場合）
  - `C:¥Program Files¥QGIS 3.xx.x¥bin¥gdalinfo`
  - OSGeo4W Shell を起動して `gdalinfo`
- macOS
  - `/Applications/QGIS.app/Contents/MacOS/bin/gdalinfo`

GDAL/OGR がリニューアル

- ここまでは、今までの GDAL/OGR の概要

- ここからが本日の本題
- GDAL 3.11（2025/05）で大幅にコマンド体系が変更

- 2025/06実施の公式ウェビナーから抜粋して紹介
  - https://download.osgeo.org/gdal/presentations/GDAL%20CLI%20Modernization.pdf
  - https://www.youtube.com/watch?v=ZKdrYm3TiBU
  - GDAL 3.12で変更された箇所は補足で説明

Everybody ❤️ GDAL utilities!

- 2021年からスポンサーシップを受付
  - 非常に多くの財政的支援があったとのこと
  - 様々な課題解消、ドキュメント整備などのため

- 2024年にアンケート
  - 非常に幅広い層の602名ものユーザから回答
  - 使用年数、取得方法、どのように GDAL を使っているか
  - よく使うファイル形式などなど
  - https://gdal.org/en/stable/community/user_survey_2024.html

Everybody 🤬 GDAL utilities!

- GDAL/OGR をより使いやすくするには？

OMG the command line options on the tools. ***Burn the village to the ground and build again****.*

- 何が彼をそんなに焚き付けたのか。。。

ここがヘンだよ GDAL/OGR

- アンダーバーがあったりなかったり
  - `gdal_translate` vs `gdalwarp`
  - `gdal_merge` vs `ogrmerge`
- 入出力の順番が不統一
  - `gdal_translate input.tif output.tif`
  - `gdalwarp output.tif input.tif`
- 上書きしたりしなかったり
  - `gdal_translate` 上書きする（勝手に）
  - `gdalwarp` 上書きしない（`-overwrite` が必要）

ここがヘンだよ GDAL/OGR

- オプション名が不統一（対象領域 / target extent）
  - `gdal_translate -projwin ulx uly lrx lry`
  - `gdalwarp       -te      llx lly urx ury`
- オプション名が不統一（出力ラスタサイズ / target size）
  - `gdal_translate -outsize width height`
  - `gdalwarp       -ts      width height`
- 似たようなオプション（対象領域の切り取りと座標の割り当て）
  - `gdal_translate -projwin ulx uly lrx lry`
  - `gdal_translate -a_ullr  ulx uly lrx lry`

ここがヘンだよ GDAL/OGR

- ひとつのコマンドで多種多様なことができる
  - 代表的なものに gdal_translate や ogr2ogr
  - ogr2ogr には76個ものオプションがある
  - このため互いに排他的なオプションの組み合わせも存在する

- 27年もの歴史があるツールの技術的負債
  - 統一されてない命名規則
  - 後方互換性を保ち続けた

- そうだ、村を焼こう

Burn the village to the ground 🔥

- 新しく `gdal` コマンドを導入
  - git 風 (`git clone`, `git commit`, ...) のサブコマンド方式
- UNIX のコマンド慣例に準拠
  - ショートオプション　`-i input.tif`
  - ロングオプション　`--output-format=GTiff`
- データセットは INPUT(s) OUTPUT 順に統一
  - `gdal vector convert input.shp output.gpkg`
- ファイルは上書きされない（`--overwrite` が必要）
- 肥大化したコマンドを、細かな機能ごとに分割

※ `gdal_translate`, `ogr2ogr` 等の従来のコマンドは存続（互換性維持）

gdal

- トップレベルコマンドは gdal
  - gdal <COMMAND> 形式。基本は gdal raster と gdal vector
  - gdal info は gdal raster info / gdal vector info のショートカット
  - 同様に gdal convert / gdal pipeline もそれぞれショートカット

```
$ gdal --help
USAGE: gdal <COMMAND> [OPTIONS]
where <COMMAND> is one of:
  - convert:  Convert a dataset (shortcut for 'gdal raster convert' ...
  - dataset:  Commands to manage datasets.
  - driver:   Command for driver specific operations.
  - info:     Return information on a dataset (shortcut for 'gdal ...
  - mdim:     Multidimensional commands.
  - pipeline: Process a dataset applying several steps.
  - raster:   Raster commands.
  - vector:   Vector commands.
  - vsi:      GDAL Virtual System Interface (VSI) commands.
```

gdal raster

- gdal raster <SUBCOMMAND>
  - gdal raster convert ファイル形式変換 (gdal_translate)
  - gdal raster reproject 再投影 (gdalwarp) など

```
$ gdal raster --help                                                    # 一部抜粋
Usage: gdal raster <SUBCOMMAND> [OPTIONS]
where <SUBCOMMAND> is one of:
  - calc:             Perform raster algebra
  - convert:          Convert a raster dataset.
  - create:           Create a new raster dataset.
  - edit:             Edit a raster dataset.
  - hillshade:        Generate a shaded relief map
  - info:             Return information on a raster dataset.
  - pipeline:         Process a raster dataset applying several steps.
  - polygonize:       Create a polygon feature dataset from a raster band.
  - reproject:        Reproject a raster dataset.
  - resize:           Resize a raster dataset without changing the ...
```

gdal vector

- gdal vector <SUBCOMMAND>
  - gdal vector convert / gdal vector reproject (ogr2ogr)
  - gdal vector grid 点データからラスタ (gdalgrid) など

```
$ gdal vector --help                                                    # 一部抜粋
Usage: gdal raster <SUBCOMMAND> [OPTIONS]
where <SUBCOMMAND> is one of:
  - clip:                 Clip a vector dataset.
  - convert:              Convert a vector dataset.
  - filter:               Filter a vector dataset.
  - grid:                 Create a regular grid from scattered points.
  - info:                 Return information on a vector dataset.
  - pipeline:             Process a vector dataset applying several steps.
  - rasterize:            Burns vector geometries into a raster.
  - reproject:            Reproject a vector dataset.
  - select:               Select a subset of fields from a vector dataset.
  - sql:                  Apply SQL statement(s) to a dataset.
```

gdal vector geom

- GDAL 3.11で導入されたが、 3.12で gdal vector に統合
  - (例) gdal vector geom buffer → gdal vector buffer
  - 公式ウェビナー資料から状況が変化
  - gdal vector geom は GDAL 3.12で非推奨、 GDAL 3.13で廃止

```
$ gdal vector --help                              # GDAL 3.12で geom 関係抜粋
Usage: gdal raster <SUBCOMMAND> [OPTIONS]
where <SUBCOMMAND> is one of:
  - buffer:              Compute a buffer around geometries of a ...
  - explode-collections: Explode geometries of type collection of a ...
  - make-valid:          Fix validity of geometries of a vector dataset.
  - segmentize:          Segmentize geometries of a vector dataset.
  - set-geom-type:       Modify the geometry type of a vector dataset.
  - simplify:            Simplify geometries of a vector dataset.
  - swap-xy:             Swap X and Y coordinates of geometries of ...
```

コマンドの例

- ファイル情報を取得 (gdalinfo, ogrinfo)
  - `gdal vector info input.gpkg`
- ファイル形式を変更 (gdal_translate, ogr2ogr)
  - `gdal raster convert --of=COG input.nc output.tif`
- 座標参照系の再投影 (gdalwarp, ogr2ogr)
  - `gdal raster reproject --dst-crs=EPSG:4326 in.tif out.tif`
- 複数のファイルをひとつの仮想ファイルに (gdalbuildvrt)
  - `gdal raster mosaic ./src/*.tif out.vrt`
- ベクタデータをひとつのファイルに (ogrmerge)
  - `gdal vector concat --mode=stack *.shp merged.gpkg`

パイプライン

- 新しい GDAL/OGR のコマンドは機能ごとに細かく分割
  - GDAL 3.12 で gdal raster 下に46つ、 gdal vector 下に28つ
- 複数のコマンドを一連に実行できるパイプラインも導入
  - ステップの区切り記号は !
- パイプラインの例

```
$ # input.gpkg を読み込み、指定したもの以外の属性情報を削除し、再投影し、
$ # ジオメトリを修復し、指定した領域で切り抜き、出力する
$ gdal vector pipeline ! ↵
      read input.gpkg ! ↵
      select fields=name,geometry ! ↵
      reproject --dst-crs=EPSG:4326 ! ↵
      make-valid ! ↵
      clip --bbox=2,49,3,50 ! ↵
      write --overwrite output.gpkg
```

Bash 補完（Linux 等）

- コマンド入力中に TAB キーを押すと補完
- 取り得る値が複数あるときに TAB TAB すると一覧表示

```
$ # コマンド名の補完
$ gdal <TAB><TAB>
convert   driver    mdim      pipeline  raster    vector    vsi
dataset   info

$ # サブコマンド名の補完
$ gdal raster <TAB><TAB>
as-features      compare          hillshade        pansharpen      ...
aspect           contour          index            pipeline        ...
blend            convert          info             pixel-info      ...
calc             create           mosaic           polygonize      ...
clean-collar     edit             neighbors        proximity       ...
clip             fill-nodata      nodata-to-alpha  reclassify      ...
color-map        footprint        overview         reproject       ...
```

Bash 補完（Linux 等）

```
$ # オプション名の補完
$ gdal raster convert --<TAB><TAB>
--append            --input-format     --output-format    --output
--creation-option   --open-option      --quiet            --overwrite
--input


$ # オプションの取り得る値の補完（出力ファイルフォーマット）
$ gdal raster convert --of=<TAB><TAB>
GTiff               GIF                HDF4Image          GSAG               ...
COG                 FITS               ISIS3              GSBG               ...
VRT                 BMP                PDS4               GS7BG              ...
NITF                PCIDSK             VICAR              KMLSUPEROVERLAY    ...
HFA                 PCRaster           ERS                WEBP               ...
AAIGrid             ILWIS              JP2OpenJPEG        PDF                ...
DTED                SRTMHGT            GRIB               MBTiles            ...
PNG                 Leveller           RMF                CALS               ...
JPEG                Terragen           WMS                WMTS               ...
MEM                 netCDF             RST                MRF                ...
```

Bash 補完 （ Linux 等）

```
$ # 出力形式が GeoTIFF のときの --creation-option (--co) の値の補完
$ gdal raster convert --of=GTiff --co=<TAB><TAB>
COMPRESS=             TILED=              SOURCE_PRIMARIES_RED=
PREDICTOR=            TFW=                SOURCE_PRIMARIES_GREEN=
DISCARD_LSB=          RPB=                SOURCE_PRIMARIES_BLUE=
JPEG_QUALITY=         RPCTXT=             SOURCE_WHITEPOINT=
JPEGTABLESMODE=       BLOCKXSIZE=         TIFFTAG_TRANSFERFUNCTION_RED=
ZLEVEL=               BLOCKYSIZE=         TIFFTAG_TRANSFERFUNCTION_GREEN=
   :                     :                   :

$ # --creation-option の COMPRESS が取り得る値の補完
$ gdal raster convert --of=GTiff --co=COMPRESS=<TAB><TAB>
NONE          PACKBITS      CCITTRLE      CCITTFAX4     LZMA       ...
LZW           JPEG          CCITTFAX3     DEFLATE       ZSTD       ...


$ # --creation-option の JPEG_QUALITY が取り得る値の範囲説明
$ gdal raster convert --of=GTiff --co=JPEG_QUALITY=<TAB><TAB>
##                       validity range: [1,100]
```

Bash 補完（Linux 等）

```
$ # 座標参照系の値の補完
$ gdal raster reproject --dst-crs=EPSG:667<TAB><TAB>
6670 -- JGD2011 / Japan Plane Rectangular CS II
6671 -- JGD2011 / Japan Plane Rectangular CS III
6672 -- JGD2011 / Japan Plane Rectangular CS IV
6673 -- JGD2011 / Japan Plane Rectangular CS V
6674 -- JGD2011 / Japan Plane Rectangular CS VI
6675 -- JGD2011 / Japan Plane Rectangular CS VII
6676 -- JGD2011 / Japan Plane Rectangular CS VIII
6677 -- JGD2011 / Japan Plane Rectangular CS IX
6678 -- JGD2011 / Japan Plane Rectangular CS X
6679 -- JGD2011 / Japan Plane Rectangular CS XI
```

- ファイル形式により異なるオプション等も一覧表示でき、ドキュメントの確認作業が大幅に軽減される

注意事項（だいじ！）

- 資料執筆時点で GDAL 3.12.0
- 現時点で機能を完全に移植できていない
  - 従来のコマンドでないとできないこともある
- 仕様が凍結されておらず、今後変更の可能性がある
  - 従来のコマンドにない機能も追加されている
  - ある意味、枯れた技術でなくなった

- フィードバックを受けつつ、最終的に3.16で凍結予定
- Community feedback needed !

GDAL で CS 立体図を作ろう(1)

- 新しい GDAL/OGR では CS 立体図だって作れるんです
- ご存知、標高から曲率 (Curvature) と傾斜 (Slope) を計算して、色を調整して、いい感じに重ねたやつ

```
$ # GeoTIFF 化した DEM5A がある状態
$ ls
FG-GML-533757-DEM5A-20250620.tif  FG-GML-533767-DEM5A-20250620.tif


$ # 複数の GeoTIFF からなる仮想ファイルを作成（gdalbuildvrt）
$ # 解像度0.2秒 = 0.000556度で半端なため、解像度不一致エラーがでる？
$ # --bbox と --resolution を指定することで回避
$ gdal raster mosaic --bbox=137.875,35.750,138.000,35.9166666666666667 ↵
      --resolution=0.0000555555555556,0.0000555555555556 ↵
      FG-GML-533757-DEM5A-20250620.tif FG-GML-533767-DEM5A-20250620.tif ↵
      dem.vrt
```

GDAL で CS 立体図を作ろう(2)

```
$ # dem.vrt を読み込んで、平面直角座標系に再投影（gdalwarp）して
$ # NoData の部分を補間（gdal_fillnodata）して、 GeoTIFF で出力する
$ gdal raster pipeline ! read dem.vrt ! ↵
      reproject --dst-crs=EPSG:6676 --bbox=-56200,-27400,-45300,-9300 ↵
          --resolution=5,5 --resampling=bilinear ! ↵
      fill-nodata ! ↵
      write dem_reprojected.tif


$ # 傾斜量を計算する（gdaldem）
$ # 計算手法は Horn と ZevenbergenThorne が利用できるが
$ # ドキュメントによると起伏のある地形に適しているらしい Horn を採用
$ gdal raster slope --gradient-alg=Horn dem_reprojected.tif slope.tif
```

GDAL で CS 立体図を作ろう(3)

```
$ # ガウシアンフィルタで平滑化する
$ # ガウシアンフィルタは size=3 or 5 は用意されているけど、それ以上はない
$ # σ = 2.0 くらいで size=9 のカスタムカーネルを作成して適用する
$ # --method=mean を指定して正規化を行う（係数の合計を1にする）
$ gdal raster neighbors --kernel="[[1,3,5,8,9,8,5,3,1], ↵
      [3,7,13,19,21,19,13,7,3],[5,13,24,35,40,35,24,13,5], ↵
      [8,19,35,51,58,51,35,19,8],[9,21,40,58,66,58,40,21,9], ↵
      [8,19,35,51,58,51,35,19,8],[5,13,24,35,40,35,24,13,5], ↵
      [3,7,13,19,21,19,13,7,3],[1,3,5,8,9,8,5,3,1]]" ↵
      --method=mean dem_reprojected.tif dem_smoothed.tif


$ # 曲率を計算する
$ # general curvature は画像処理におけるエッジ検出のカーネルと同じなので
$ # 用意されている "edge1" を使ってから、 1/L² スケーリングしてもよい
$ # --method=sum を指定して正規化を行わない（係数の合計がゼロなので）
$ gdal raster neighbors ↵
      --kernel="[[0,-0.04,0],[-0.04,0.16,-0.04],[0,-0.04,0]]" ↵
      --method=sum dem_smoothed.tif curvature.tif
```

GDAL で CS 立体図を作ろう(4)

```
$ # こんな感じのカラーマップファイルを用意する ( 最小値が黒、最大値が白 )
$ cat colormap_black-white.txt
100%     255 255 255
  0%       0   0   0


$ # 標高 / slope / curvature のデータラスタからカラー画像ラスタを作成
$ gdal raster color-map --color-map=colormap_black-white.txt ↵
      dem_reprojected.tif colored_height.tif

$ # 同様に、それぞれのカラーマップでカラー画像ラスタを作成
$ gdal raster color-map --color-map=colormap_white-red.txt ↵
      slope.tif colored_slope_red.tif
$ gdal raster color-map --color-map=colormap_white-black.txt ↵
      slope.tif colored_slope.tif
$ gdal raster color-map --color-map=colormap_blue-white.txt ↵
      curvature.tif colored_curvature_blue.tif
$ gdal raster color-map --color-map=colormap_blue-yellow-red.txt ↵
      curvature.tif colored_curvature_blue-red.tif
```

GDAL で CS 立体図を作ろう(5)

```
$ # 今回は真っ白の画像も用意しておく
$ gdal raster color-map --color-map=colormap_white.txt ↵
      dem_reprojected.tif colored_white.tif


$ # 各画像をブレンドして CS 立体図を作成する
$ # 今回は白画像をベースに、不透明度を指定しつつブレンド
$ # 不透明度を指定したカラーマップを使って RGBA 画像を作成する方法もある
$ # 色やブレンド比率の細かな調整はしておらずパラメータは適当（PoC）
$ gdal raster pipeline ! read colored_white.tif ! ↵
      blend --opacity  5 --overlay colored_height.tif ! ↵
      blend --opacity  5 --overlay colored_curvature_blue.tif ! ↵
      blend --opacity 15 --overlay colored_slope_red.tif ! ↵
      blend --opacity 50 --overlay colored_curvature_blue-red.tif ! ↵
      blend --opacity 25 --overlay colored_slope.tif ! ↵
      write csmap.tif
```

GDAL で CS 立体図を作ろう(6)

完成図（ QGIS 上で国土地理院 淡色地図に重ねた例）