CRS and/or Tile as dimensions in OpenEO datacubes

Edzer Pebesma

January 7, 2020

Abstract

Datacubes are an attractive data model for handling satellite image collections, and are used as such in the OpenEO API. They assume however that a unique mapping exists from spatial dimensions (x and y) to locations on the Earth. This requires that a single coordinate reference system (CRS) is available to uniquely translate between x and y and Earth locations. When image collections involve sets of tiles spanning several CRS (such as Sentinel-2: tiles by UTM zone), this no longer works: the mapping from x and y dimension indexes to Earth coordinates depends on the UTM zone. One way out of this is to warp data (at some stage) to a common CRS, but this involves image resampling and data loss. Another approach, discussed here, is to distribute images from different CRS over a new dimension, the CRS dimension. Given that idea, the same can be done for image tiles. This note discusses the implications of this approach.

1 Datacubes

Datacubes are arrays, defined as

$$A : D \rightarrow V$$

that map from n dimensions $D = D_1 \times D_2 \times ... \times D_n$ to p attribute values $V = V_1 \times ... \times V_p$, and individual dimensions D_i are finite and totally ordered. Implementations of the array model typically represent dimension values as integers. Due to the finiteness property, however, there always exists an invertible dimension transformation function that relates the array index integer number to the original dimension value (Lu et al, 2018). In OpenEO, we made the simplification that $p = 1$, i.e. we have only a single (scalar) attribute value a in each single cell:

$$D[i, j, k, ...] = a$$

with $D[i, j, k, ...] = \{D_1[i], D_2[j], D_3[k], ...\}$ For spatial coordinates, say dimension 1 and 2, a coordinate reference system (CRS) specifies how to translate between spatial coordinates ($D_1[i] = x$ and $D_2[j] = y$) and longitude/latitude

values. (Note that all CRS for which graticule lines are not vertically and horizontally aligned straight lines do not have a unique translation between x and longitude, or y and latitude, but require coordinate pairs.)

2 UTM zones

If we have data distributed over different UTM zones, sets of identical coordinates coming from different UTM zones refer to different places on the Earth. With the PROJ[1] utility `cs2cs` we can illustrate this:

```
$ echo 288776 9110728 | cs2cs "+proj=utm +zone=25 +south" +to "+proj=longlat"
34d54'59.729"W 8d2'25.882"S 0.000
$ echo 288776 9110728 | cs2cs "+proj=utm +zone=25 +north" +to "+proj=longlat"
46d26'5.735"W 81d49'55.903"N 0.000
$ echo 288776 9110728 | cs2cs "+proj=utm +zone=33 +north" +to "+proj=longlat"
1d33'54.265"E 81d49'55.903"N 0.000
```

Clearly, this no longer allows a unique translation from x and y to longitude/latitude. This means that we need to have the combination (x, y, CRS) to uniquely translate from and to longitude and latitude, and so UTM zone becomes a dimension.

Datacubes of this form ($x \times y \times \text{CRS} \times ...$) are special in the following sense:

- the "spatial" dimensions x and y have no CRS: their values (e.g. $x = 288776$ and $y = 9110728$) are only meaningful (can be related to world coordinates) in the context of the CRS slice they are in.
- a single slice along the CRS dimension gives a spatial data cube with CRS
- the process `resample_spatial` could resolve ("reduce") the entire CRS dimension, and creates a datacube with spatial dimensions that have the target CRS
- alternatively, `reduce_crs` could reduce the CRS dimension and require a target CRS to reduce (warp) the CRS-varying slices into.
- spatial selections would deselect entire CRS slices first, then mask out particular regions for remaining CRS slices
- processes like `aggregate_polygon` might have to select, then warp

3 Tiles

An activity similar to having CRS as a dimension is to have TILE as a dimension. One could actually interpret e.g. the Sentinel-2 tiles (found at https://sentinel.esa.int/web/sentinel/missions/sentinel-2/data-products) as

[1] http://proj.org/

special case of CRS: in addition to the UTM zone, they have the origin of the tile in the UTM zone considered. Doing this would remove the need to mosaic tiles that are in a common CRS.

4 Why go through this length?

The use case motivating this note is one where, over a large spatial region (crossing UTM zones) spectral and temporal dimensions were reduced (e.g. to create phenology or crop productivity measures), but where the results are needed at the original pixel level, that is without warping or resampling any rasters. For this case, only the temporal and spectral dimensions are processed, and anything spatial remains untouched.

In this workflow, the steps are basically: for every tile, do:

1. reduce time and spectral
2. write back tile

This is not what you would typically do with Google Earth Engine, but for those in the habit of creating products at native resolution, it may well be.

References

- Meng Lu, Marius Appel, Edzer Pebesma, 2018. Multidimensional Arrays for Analysing Geoscientific Data. ISPRS Int. J. Geo-Inf. 2018, 7(8), 313; https://www.mdpi.com/2220-9964/7/8/313/htm