The authors perform a comparison of several systems on 3D and 4D gridded data, with a particular view on parallelization. Tools addressed are THREDDS/NetCDF on standard file system and an object store, pangeo/Dask, and Hadoop/Spark/Parquet. The conceptual model is a 4D cube where the vertical dimension is "nullable" in case of 3D data. This resembles the model of the Barrodale engine on top of Informix, for example.

In Section 2, I get puzzled about the experimental setup description:

- the "criteria identified to stress" are not justified and sometimes surprising: why is it relevant for storage (!) whether a pixel is ocean or land? why is complex processing, heterogeneous data fusion, etc. not considered?

We first thank the reviewer for its thorough review of the paper and the detailed analysis and comments.

We are now clarifying in the manuscript the criteria and the choice of datasets.

- "enrichment of CSV locations": CSV = comma-separated values? if so, why is that format choice important over, e.g., JSON? what does enrichment mean, and what is the scenario? why is it relevant?

Explanations of the enrichment and references are added to the manuscript

- "parallelized processing" (check language use!) is not a user scenario, but an implementation detail. What would be interesting is what operations exactly to parallelize - for example, an edge filter or matrix operations are harder to parallelize than the trivial pixel operations such as NDVI or subsetting.

Statistics and trends detections are performed with parallelization. The idea is to prove the efficiency of the proposed storages, compared to traditional THREDDS solution.

Generally there seems to be a lack in concise description of the relevant choices, impacts, and measures. Just one example (p 11): "We repeated the tests several times to obtain the most reliable metrics." A rigorous approach might "run all tests 5 times on a [hot|cold] setup, discarding the maximum and minimum value and averaging the 3 remaining measurements".

The following precision was added: We repeated the tests three times to get a more reliable measurement of performances

The algorithms used for the scenarios, such as extracting significant continuous areas and enrichment, are only given cursorily, without a rigorous pseudo code or mathematical description.

Notebooks used to test the Pangeo implementation were added in github, https://github.com/clstoulouse/ParquetCubeNotebooks and referenced in the paper.

In 3.3, it is claimed that datacubes need reprojection which should be avoided. However, for join/fusion of datasets in different projections there needs to be a reprojection. And rescaling (which likewise involves resampling) is performed routinely by the approach. So I do not see the substantial difference. In fact, substantial preprocessing takes place at datacube creation time, massaging data to make them suitable for the processing later on. For example, all cubes are forced into the same space/time resolution - a brute-force method, and certainly more dangerous from a scientist perspective than reprojection. Other datacube approaches work on the original data and perform dynamic recombination.

Could you please precise why it is dangerous for a scientist to use a library to process the data the way he expects? It is true that the Rasdaman implementation makes it easier to access the data and that such ingestion mechanism facilitates the use of a language to manipulate the data. With this new Parquet datacube, we describe an alternative based on another paradigm.

Reading the datacube manifesto, https://earthserver.eu/tech/datacube-manifesto/The-Datacube-Manifesto.pdf, we can see that the Datacube terminology used for the Parquet datacube described here could not be compliant with the last requirement: “Datacubes shall support a language allowing clients to submit simple as well as composite extraction, processing, filtering, and fusion tasks in an ad-hoc fashion”.

It is true that the Spark access to the Parquet Cube developed by CLS is not compliant with this 6th requirement of the datacube manifesto, but the Pangeo implementation which allows the SQL like selection in a dataframe initialized from the parquet files makes the Parquet Cube complaint with it.

The Parquet format does not seem efficient for cubes. By materializing the coordinates for each point the data volume is blown up immensely, and processing (including data bus transport between RAM and CPU) get slowed down. Combine this with the 3x explosion contributed by HDFS (not to speak about its inflexible page size), it is not clear how this approach can be efficient in comparison to others published. Certainly not a "green computing"!

The results of this paper are comparing the native NetCDF files size with the Parquet Cube implementation. We did not see the volume blowing up immensely doing our research.

Partitioning, a well-known technique for gridded data, is applied here as well. The parameters chosen are not justified, though: why regular partitioning? why partition length 1 day along time? We just learn "the partitioning-per-day makes sense",probably because the demonstration scenarios have been pre-trimmed to that. This will be problematic in a general-purpose operational deployment.

You are right, we did not stress the Parquet Cube implementation with the partition size and this limitation is mentioned in 595

Sadly, the performance comparison of the different implementations was done on rather different infrastructure, so comparison of the results is problematic.

This limitation is now clarified and commented in the manuscript. The objective of this paper is to explore and demonstrate that the alternative using the Parquet format has some strengths and weakness compared to others but nonetheless interesting for specific users and use cases.

In Table 3, performance results for single-file conversion are reported. There is a breathtaking span from 1 to 12816 seconds per file. Unfortunately, it is not explained sufficiently.

Indeed some measurements were not performed for all datasets and implementations that is why there are not applicable (N/A) values present in table3..The input files were too big for the THTREDDS implementation and unfortunately, the Pangeo implementation did not log the ingestion time. We do not think it was relevant to detail in the paper as it does not change any of our conclusion or hypotheses.

Performance results (Figure 8) are a little hard to follow due to nonlinearity - eg, time axis extraction starts with 1D (incidentally the stored grid resolution) and then scales with a factor 30 (?), 3, 2, 2. Equidistant spacing would help understanding the results.

Comment are added:

In figure 8 below, time units are D for days, M for months, Y for year.

The linearity in time for a given geographical region can be observed for the Pangeo/Zarr or Spark parquet implementations, except when the initialization of the context request of the subsetting is dimensioning compared to the subsetting process itself.

Surprisingly, performance degrades quadratic with increasing data volume returned, whereas other tools in the field commonly show a linear behavior. Parallelization does not seem to help much.

Unfortunately, we did not find yet the explanation for this quadratic effect.

Also surprising (and unexplained) is the non-monotonicity in Fig 11 for THREDDS / North Sea while THREDDS / global shows reasonable monotonicity. Further, in Fig 13 there is a huge outlier for THREDDS / hourly / 1M - is this a measurement issue, or a real result? Unfortunately, no explanation is attempted.

The stride bias mentioned in 454 is described in 255. We did not repeat it in the aforementioned paragraph.

On p 18 I would like to understand how "significant" data are determined algorithmically - as it stands, the system load generated cannot be estimated. Further, as "continuous" areas are retrieved there is likely some kernel operation involved. How does kernel computation work at partition boundaries, is it still correct (ie, fetches values from neighboring partitions where necessary)?

Significant means here with a physical value not a default value. We made the following clarification: The idea was to compare the incidence of the data type: significant continuous

physical values over wide areas (seas) versus lot of small data spots and long series of default values (lakes).

Looking at the results on p 19: Performing a simple subsetting returning an estimated 5 MB of data using 10 cores in 30 seconds is breathtakingly slow - other systems can do that in less than 1 second.

The 30 second measurement is mainly due to the initialization of the Spark Context. It was mentioned in 535 but you are right, we have also clarified this in this paragraph of the manuscript.

Overall, seeing the data set in the conclusion is described as having 2 TB and Spark/Dask were used on 50/40 cores: that means about 40 GB per node - loading that into RAM of each node and using just numpy etc. should be faster by orders of magnitude. Shouldn't the test go well beyond the cumulated RAM?

Sorry, I think there is a confusion here. The typical configuration used for the scenario 1 in the CNES HPC for Pangeo is 10 cores and 4 Gb of memory as mentioned. The tests were run in an environment quite easy to reproduce for readers and not expensive.

Bottom line, what I take home is

- THREDDS is not really scalable (which is confirmed by other studies)

- Parquet works well in stuations where data and scenarios are carefully aligned

- benchmark deployments have so many special tweaks and differences that a comparison is difficult

- both Spark-Parquet and Pangeo-Zarr fall significantly behind the performance of other tools around

- dynamic partitioning (as studied by Paula Furtado, for example) has not yet found wide recognition

OK. We updated the paper and the conclusion to focus on the feasibility aspect more than on performances.

editorial comments:

- p 3: URLs in the make the text ugly to read, better make it a reference.

OK, done

- p 3: "N variables" - what does that want to tell us? Unknown, variable, or...?

Ok, the numbers of variables are now mentioned in the paper.

- maybe recheck for typos, such as p 4 "librairies"
Fixed

- best use uniform nomenclature, not "4G" and "4 go" for 4 GB

OK done

- p 5: "The number of cores is increased for the Pangeo-Zarr and the Spark-Parquet environments." ...why? what is the number of cores there? Best motivate such decisions.

This motivation have been added to the manuscript : “to observe their horizontal scalability”

- Figure 9: as the diagram lines are greyscale they are not easy to distinguish. If color is not possible consider dashed lines etc.

Bold police and number format were adjusted to make the figures more readable.

- check for French words occurring, such as "Novembre"

Done

Citation: https://doi.org/10.5194/gmd-2021-138-RC2