NFDI4Earth

Deliverable D2.5.6

Summary of software contributions to advancing tools in the Earth System Sciences

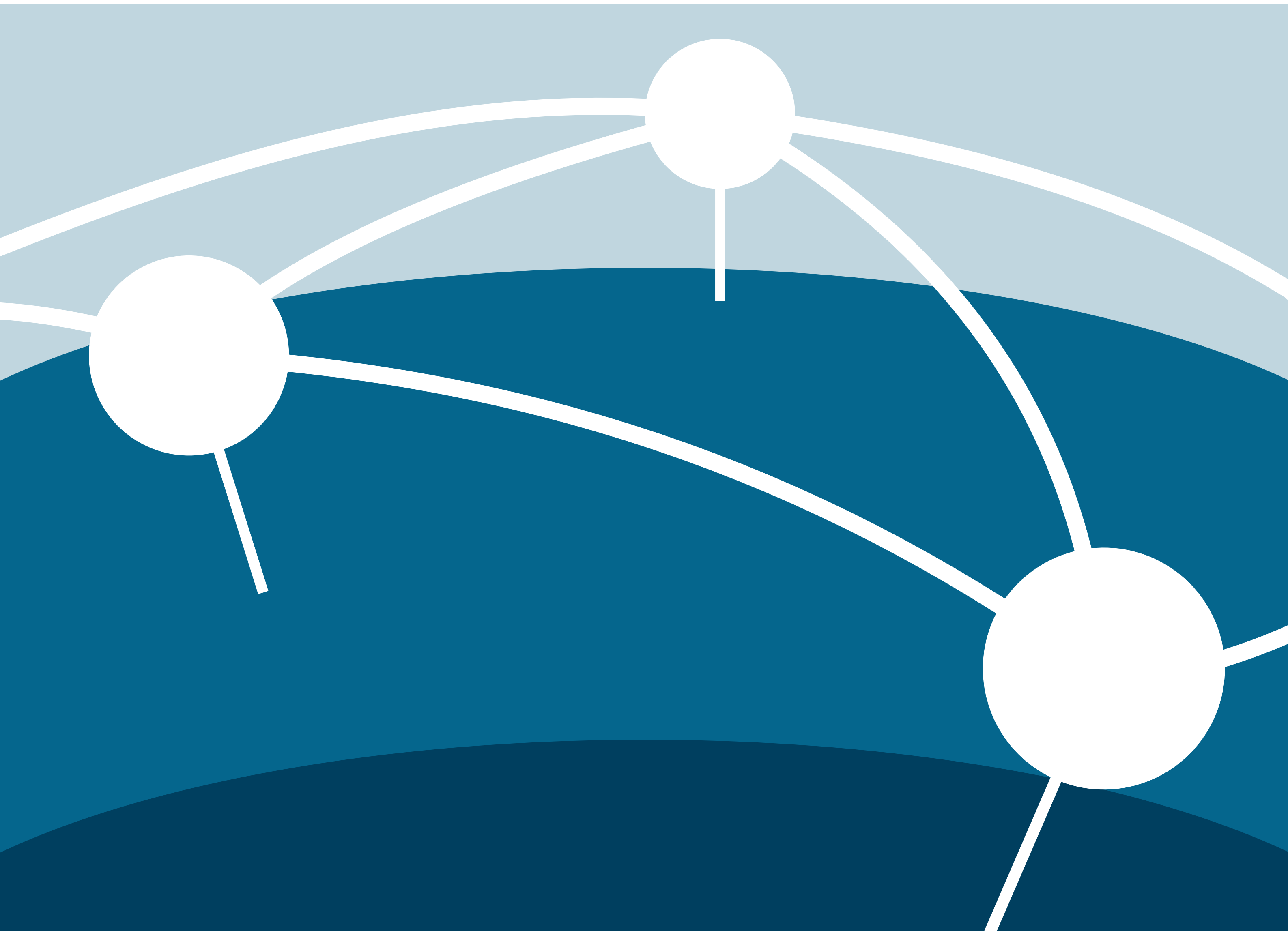
Action 4

Felix Cremer (fcremer@bgc-jena.mpg.de), Fabian Gans, Edzer Pebesma

2023-06

DOI: 10.5281/zenodo.8192731

nfdi4earth.de



Citation

Felix Cremer, Fabian Gans, Edzer Pebesma. 2023. *Summary of software contributions to advancing tools in the Earth System Sciences (NFDI4Earth Deliverable D2.5.6)*. Zenodo. https://doi.org/10.5281/zenodo.8192731

License

Acknowledgement

This work has been funded by the German Research Foundation (DFG) through the project NFDI4Earth (DFG project no. 460036893, https://www.nfdi4earth.de/) within the German National Research Data Infrastructure (NFDI, https://www.nfdi.de/).

Executive summary

This document lists all contributions to open source solutions for the handling and analysis of large gridded datasets, that have been implemented in the frame of the NFDI4Earth Measure 2.5 until end of Q2 2023. Section 2 lists the most important software contributions for different packages with a small description of the aim of every package. Section 3 lists other contributions to the open source community and section 4 lists the contributions to documentation and tutorials.

Contents

1 Software Contributions

In the first year of the NFDI4Earth project, M2.5 put its focus on advancing tools in datacube technologies. In the open source world, innovative technological solutions are usually developed by a small group of people building solutions for a set of problems in their research area. When these developed solutions prove to be successful they get picked up by other communities and get generalised and formalised to work in a wider set of applications. As these new technologies are not formalised and fully specified from the start this can cause serious interoperability problems because different implementations have different extensions and interpretations of the technology specification.

For example the Zarr storage format was initially developed by neuroscientists to store multidimensional MRI data while allowing for efficient cloud storage and access as well as fast parallel read and write operations. Afterwards the storage format was applied in other fields like geoscience to store spatiotemporal data cubes and backends were developed for analysis tools like xarray and dask to access and process this data, making them accessible and interoperable with other datasets. In this process the first large datasets were published, in particular the Pangeo community spearheaded the distribution of large datasets like CMIP6 in the Zarr format on cloud storages.

While these developments happened in Python only, most of the data analysis is done in high level programming languages like Python, R and Julia or using interfaces in these languages for models in lower level programming languages like C or Fortran. This means that there is currently an accessibility barrier for scientists relying on workflows written in other programming languages or the depending on tools like GDAL written in low-level languages.

In order to close this gap and to enable better data handling in different programming languages we worked on different Julia and R software packages to remove these access barriers and make these datasets accessible and interoperable with data from other sources in these different programming languages. Finally, we participated in the standardisation of the Zarr data format and Fabian Gans is member of the Zarr Implementation Council. We also participated in the Pangeo community to enable the documentation of good data handling practices.

The software and community contributions are listed per package with a short description of the purpose of the packages.

1.1 YAXArrays.jl

YAXArrays.jl is an open source Julia library for large-scale computations on gridded datasets. These datasets can be larger than the RAM memory of the computer so that they need to be loaded from disk in smaller chunks. YAXArrays.jl provides access to a variety of file formats like Zarr, NetCDF, GeoTiff and more. YAXArrays.jl enables large distributed computations by applying user-defined functions to huge datasets in an efficient way by respecting the underlying chunking structure of the input datasets. It automatically supports multi-threaded and distributed computations or a mix of both, depending on the cluster architecture. It tries to mimic the data model of the xarray Python library, so that dimension names form the basis of merging broadcast operations. It is a computational framework on top of the DiskArrays.jl package and uses Zarr.jl, NetCDF.jl and GDAL.jl for the access of different data formats.

In the first year of the NFDI4Earth project the internal representation of the data has been changed to use the DimensionalData.jl package. This change enhances the interoperability of different data cube solutions in Julia. Besides this large change, multiple pull requests with bug fixes and performance improvements as well as contributions to the documentation have been worked on.

1.2 Zarr.jl

We contributed to both the Zarr specifications and to improvements of the Julia Zarr implementation. The improvements include a change in the interpretation of fill values to be compliant with the Zarr specifications and to match other implementations, performance improvements for reading data from web resources as well as reviewing several pull requests with bug fixes and documentation improvements.

1.3 Zarr Reading and Writing in R with `stars`

The R package `stars` has been augmented with functions `read_mdim()` and `write_mdim()` that read and write multidimensional arrays using GDAL's C++ API for multidimensional arrays. This API currently supports reading and writing NetCDF and Zarr files, and requires GDAL version 3.5. It supports partial reads of large data cubes by limiting the extent and resolution on each dimension.

1.4 PyramidScheme.jl

PyramidScheme.jl is a Julia package for the computation of image pyramids for large gridded datasets. The image pyramids can then be used for a fast and interactive visualisation of the data. It currently works for the computation of two dimensional aggregations with user-defined aggregation functions.

The PyramidScheme.jl package has been developed in the scope of the NFDI4Earth project and a first working prototype is available as open source software at https://github.com/JuliaData

Cubes/PyramidScheme.jl. A next step is to enable read and write to the emerging metadata standards for the handling of pyramid data information so that the image pyramids can be exchanged between different programming languages.

1.5 Other R packages

The R package `terra` has also undergone strong developments recently, and supports reading and writing of datacubes both using the NetCDF library directly or using the GDAL MultiDimensional Array C++ API. The package represents data in `SpatRaster` objects, which can hold up to three-dimensional datacubes.

2 Community Participation

2.1 Zarr-specs

The Zarr format is a community-driven data format as described in Deliverable D2.5.1. The current version of the Zarr specification (v2) has seen wide adoption in different communities (Neuroscience, Climate and Earth System Sciences) as a cloud-compatible alternative to HDF5. One of the members of M2.5 (Fabian Gans) became member of the Zarr Implementation Council with contributions to the discussion on further improvements on the Zarr specification. During the last reporting period we helped shaping the upcoming v3 of the Zarr specs in several discussions on GitHub, in particular by emphasising interoperability of the standard across different programming languages and contributed to the final review of this new version of the data format.

2.2 Workshop on raster and vector data cubes in Python, Julia and R

The vector data cube is a progression of the data cube concept on data with arbitrary underlying geometries see https://www.r-bloggers.com/2022/09/vector-data-cubes/ for an introduction of the idea. To enable the exchange of data sets which have been generated in different programming languages it is important to agree on data standards for the storage of such datasets. A workshop on the implementation of vector data cubes in Python, Julia and R will be held in September 2023 at the Institute for Geoinformatics in Münster. Handling vector data cubes is one of the focus areas of that workshop.

2.3 Pangeo-Europe

We participate in the Pangeo-Europe community to learn further about data handling challenges in different communities of the Earth System Sciences which have to deal with large data. In the scope of the Pangeo community we participate in a C-SCALE use case to test the bring your algorithm to the data approach with the YAXArrays.jl package. The C-SCALE project enables working with Copernicus data in the European Open Science Cloud.

3 Tutorials and Documentation

We disseminated the use of the data cube solutions that we are developing by improving the documentation and also by providing online and offline tutorials like the Julia EO Workshop and the Pangeo Show & Tell The recordings of these tutorials are available for future use.