General guideline – Compatibility between xarray and dask

- Xarray has the datatype of DataSet, DataArray, and Coordinates
  - In some numpy-inherited operations, e.g., interp, DataArray need to be turned into DataSet, thus **DataArray from a coordinate of DataSet cannot be operated**, since variable names need to be different from coordinate names.
  - **DataArray from a coordinate of DataSet cannot be chunked**, workaround: save to .nc and read in and then 'chunk'. (**Note .load() before .chunk, to avoid file busy!**) Unknown reason.
  - Guessing: a coordinate of DataSet is a special type of DataArray that is actually not fully equivalent to DataArray and thus not compatible.
  - List of 1 element will become scalar after saved and read from files. e.g., shape of 1D vector is no longer an iterable to be used in np.reshape().
- Dask and Numpy
  - https://docs.dask.org/en/latest/array-numpy-compatibility.html
    Many are direct(ufunc), i.e., a bit faster than for loop, similar to apply_ufunc
  - Numpy cannot interpolate datetime correctly; in Xarray+Dask, **the variables will be automatically dropped**.
  - Numpy-inherited operations, e.g., interp, will explicitly compute without using assigned Dask chunk.
  - Some operations remain lazy in if statement, e.g., .any(), so before .item(), **it requires .compute() to use it in if statement**.
  - To compute, could be .compute() or .persist() (??)
- Xarray and Dask
  - map_blocks learns/**needs the shape of the whole target DataSet/DataArray at the beginning**, so if the function has changing dimension size, the template need be assigned
  - apply_ufunc is more designed for a numpy array than Xarray datatypes

Use Xarry+Dask+SVD for large dataset – 1 – basics

- Documents for a brief understanding:
  - Intro to Xarray & netcdf: https://docs.xarray.dev/en/stable/getting-started-guide/quick-overview.html
  - Dask alone: https://examples.dask.org/array.html ; https://docs.dask.org/en/stable/10-minutes-to-dask.html
  - To understand how to maximize Dask features, you need to learn chunks, when to rechunk/load/compute: https://docs.dask.org/en/latest/array-chunks.html
  - Xarray+Dask: https://docs.xarray.dev/en/stable/user-guide/dask.html; https://tutorial.xarray.dev/intermediate/xarray_and_dask.html
- Further understanding: my own experiences
  - Xarray & netcdf: Learn about assigning coordinates, inheriting attributes, datatype changes
  - Dask(Xarray+Dask) and Numpy is not very compatible, check compatibility: https://docs.dask.org/en/latest/array-numpy-compatibility.html
  - If not compatible, try map_blocks (or apply_ufunc), learn to use template: https://docs.xarray.dev/en/stable/user-guide/dask.html#map-blocks ; https://tutorial.xarray.dev/advanced/map_blocks/simple_map_blocks.html
- Full reference websites (showing example pages):
  - Xarray: https://docs.xarray.dev/en/stable/generated/xarray.open_dataset.html
  - Dask: https://docs.dask.org/en/latest/generated/dask.dataframe.read_hdf.html

Use Xarry+Dask+SVD for large dataset – 2 – usage

- The useful (readable) reference links
  - https://blog.dask.org/2020/05/13/large-svds
- Tools:
  - Tutorial example: https://examples.dask.org/machine-learning/svd.html
  - Cheap (randomized) SVD, svd_compressed: https://docs.dask.org/en/latest/generated/dask.array.linalg.svd_compressed.html
  - Dashboard monitoring (with OSCAR OOD): https://docs.dask.org/en/latest/dashboard.html
- Other reference websites:
  - There are unavoidable issues due to Xarray/Dask not fully developed
  - Configuration setup (e.g. set temporary directory for memory spill): https://docs.dask.org/en/latest/configuration.html#dask.config.set
  - Further save memory by compute=True: https://github.com/dask/dask/pull/5041
  - Manual workaround for conflicting chunks when saving: https://github.com/pydata/xarray/issues/5219

More about Dask memory managing

- Dask worker/client management could be either through command line or in-program assignment
  - https://distributed.dask.org/en/stable/worker.html (e.g. multi-thread+1process v.s. 1thread+multi-process)
  - https://docs.dask.org/en/latest/scheduling.html
    https://docs.dask.org/en/stable/scheduler-overview.html
- The client creates a scheduler and then decide the tasks to do, stored in a graph
  - Balance between task numbers and memory size -> choose appropriate chunk size
  - Healthiness, check dashboard: https://docs.dask.org/en/latest/dashboard.html
  - OSCAR & dashboard using local host (from Ali Siddiqui): https://github.com/asiddi24/Software_hacks/blob/main/oscar_dask_dashboard.md
- Worker memory behavior
  - https://distributed.dask.org/en/stable/worker-memory.html
  - Avoid unmanaged, avoid a lot of IO between processes, avoid a lot of spilling (slow down and possibly not correctly managed)
  - Spill directory: dask.config.set({'temporary-directory': XXX})