Xarray-DataTree: Hierarchical Data Structures for Multi-Model Science

Or What Trees Can do for You

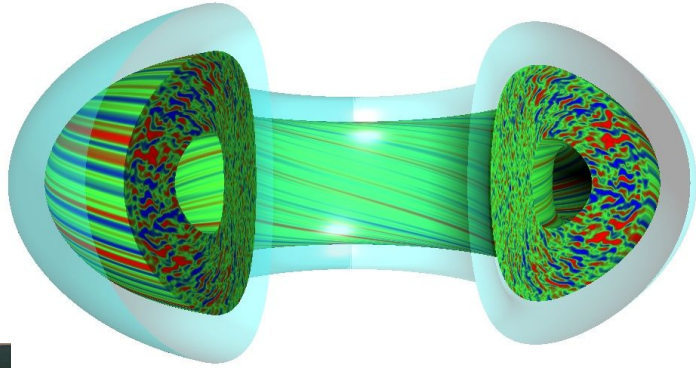
Thomas Nicholas*,
Julius Busecke

Lamont-Doherty Earth Observatory
COLUMBIA UNIVERSITY | EARTH INSTITUTE

*thomas.nicholas@columbia.edu
*github.com/TomNicholas

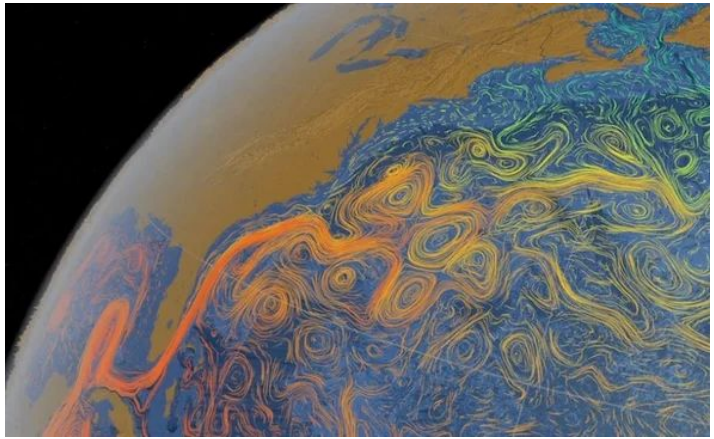
Who am I?

- Oceanographer (ex-plasma physicist)



- Xarray core dev & Pangeo user

- I like tools that all fields of science can use…



- My first AMS! 😁

PANGEO

Talk overview

1. Problem of Data Complexity

2. Xarray-DataTree can help!

3. Examples of people using DataTree already

Complexity of Big Data

- Big data has many challenges:
  - Compute / scalability … (use dask etc.)
  - Storage / IO / archiving … (use Zarr etc.)
  - But also complexity!

E.g. searching for some CMIP6 data…

```
[3]: cat = col.search(experiment_id=['historical', 'ssp585'], table_id='Oyr', variable_id='o2',
                      grid_label='gn')
     cat.df
```

[3]:

	activity_id	institution_id	source_id	experiment_id	member_id	table_id	variable_id	grid_label	
0	CMIP	CCCma	CanESM5-CanOE	historical	r1i1p2f1	Oyr	o2	gn	gs://cm
1	CMIP	CCCma	CanESM5-CanOE	historical	r2i1p2f1	Oyr	o2	gn	gs://cm
2	CMIP	CCCma	CanESM5-CanOE	historical	r3i1p2f1	Oyr	o2	gn	gs://cm
3	CMIP	CCCma	CanESM5	historical	r10i1p1f1	Oyr	o2	gn	gs://cmip6/CMIP/CCCm
4	CMIP	CCCma	CanESM5	historical	r10i1p2f1	Oyr	o2	gn	gs://cmip6/CMIP/CCCm
...	...	...	...	...	...	...	...	...	
133	ScenarioMIP	IPSL	IPSL-CM6A-LR	ssp585	r4i1p1f1	Oyr	o2	gn	gs://cmip6/Sc

But so much going on!

Science has lots of complicated data

- Data at multiple resolutions
- Many parameters
  - e.g. CMIP models, scenarios, experiments, baselines...
- Heterogeneous data
  - e.g. observations + simulations
- Anytime your netCDF file has multiple groups

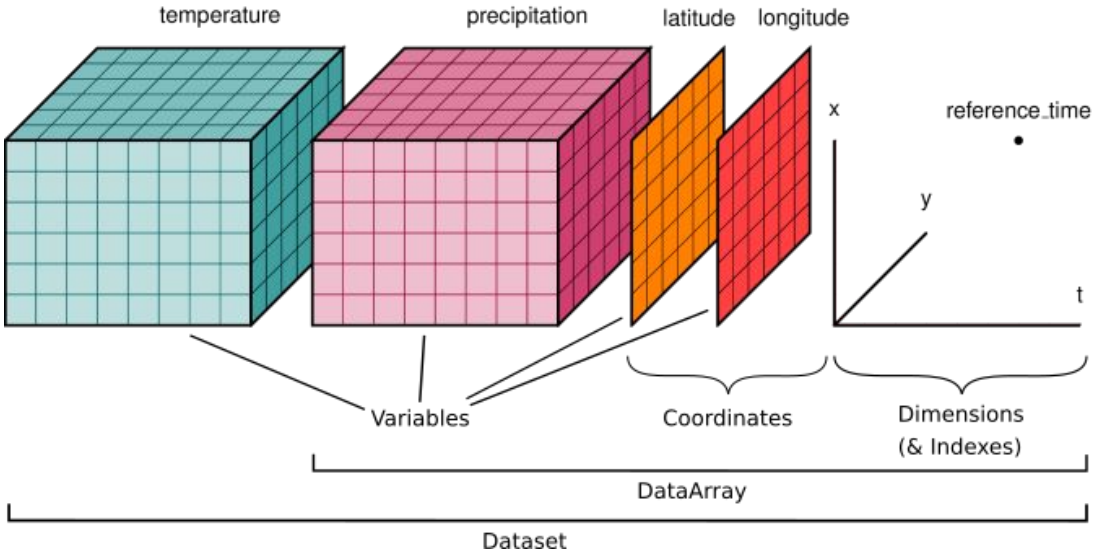
What is xarray?

- Python package providing labelled data structures
  - Wraps numpy-like arrays

- `axis=0` vs `dim='time'`

- In-memory representation of a netCDF group

- Scales via dask to TeraBytes of data!



Have you done this?

- Who here uses many separate `xarray.Dataset` objects?

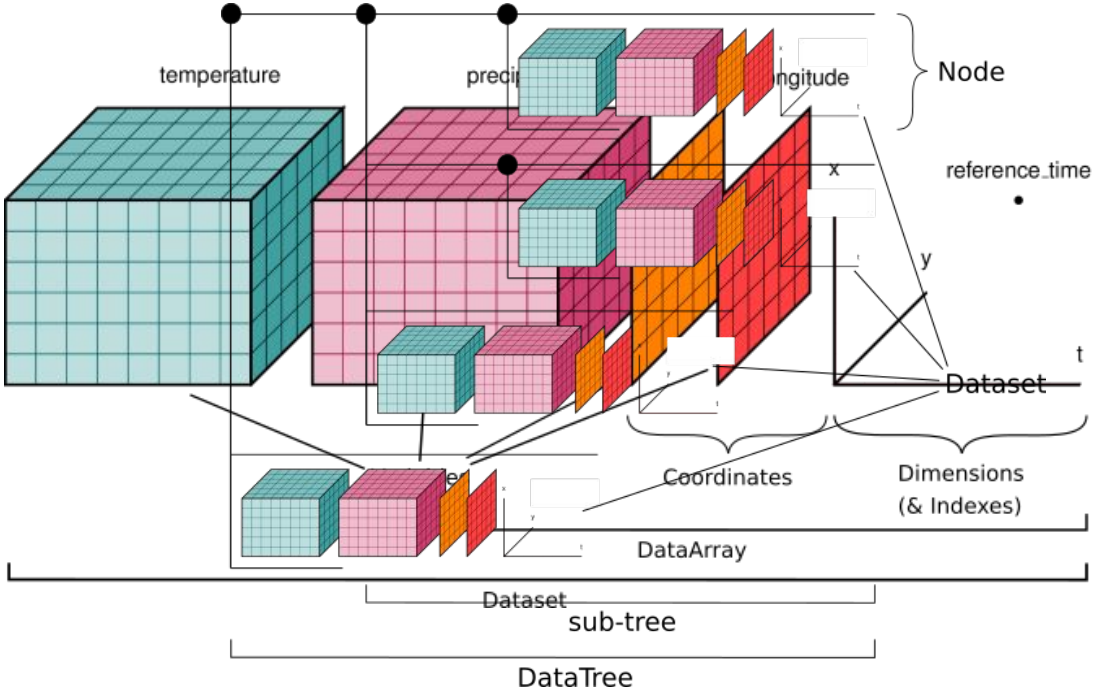
```
for key, ds in datasets_dict.items():
    avg = ds.mean(dim='time')
```

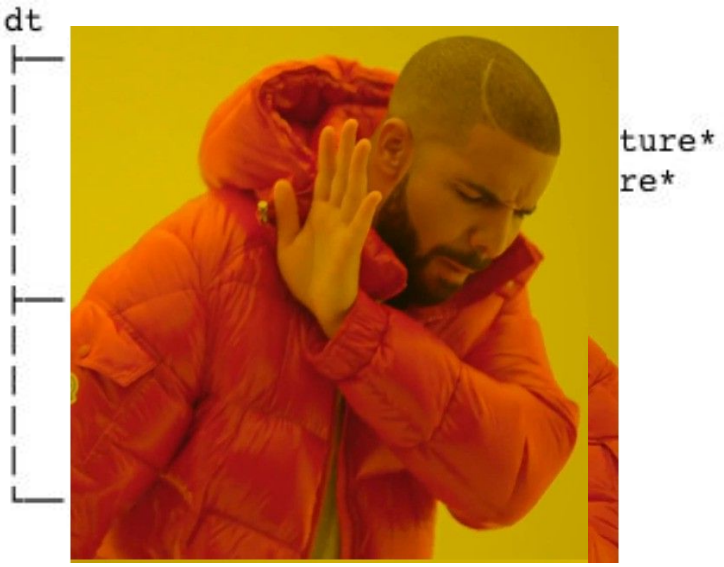
```
timeseries_obs = ds_obs.mean(['lon','lat'])
timeseries_reanalysis = ds_reanalysis.mean(...)
timeseries_model = ds_model.mean(...)
```

- Issue: real use cases are too complicated for one Dataset,
- I hope to banish this (anti-)pattern using DataTree!

Enter xarray-DataTree

- A “DataTree” is a hierarchical tree of xarray Datasets





Features 1: I / O

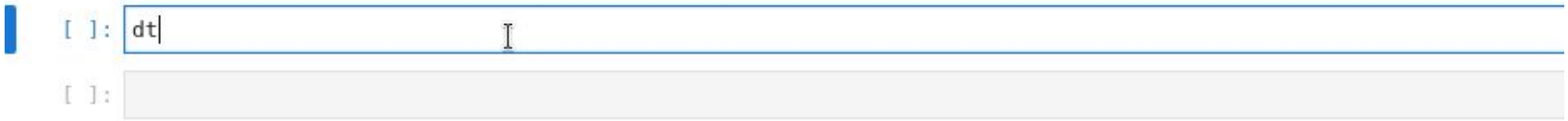
- Open any netCDF file (/ Zarr store) containing multiple groups as a nested tree

- (Can save back out as file with multiple groups too)

```
In [20]: dt = open_datatree('./datafiles/epm044463.nc')

In [21]: print(dt)
DataNode('root')
    Dimensions:  ()
    Data variables:
        *empty*
    Attributes: (12/15)
        generator:              IDAM PutData VERSION 1 (Apr 20 2021)
        Conventions:            Fusion-1.1
        class:                  analysed data
        title:                  EFIT++ Equilbrium Reconstruction epm
        date:                   2021-07-20
        time:                   13:51:58
        ...                     ...
        comment:                magnetics
        runcmd:                 efit++ -p 44463 -t mastu --pass_number 0 --runmod...
        executableGitCommitId:  c9c55b35e653f64af97db6f4a06e0181898b7a9b
        executableGitRepo:      https://git.ccfe.ac.uk/EFIT_PP/efitpp.git
        runScriptsGitCommitId:  922855671166ea5b8aac2c95858cb9cedf5795d7
        runScriptsGitRepo:      https://git.ccfe.ac.uk/EFIT_PP/efitpp.git
└── DataNode('epm')
        Dimensions:                   (time: 56)
        Coordinates:
          * time                      (time) timedelta64[ns] 00:00:00.020000 ... 00:0...
        Data variables:
            equilibriumStatusInteger  (time) int32 ...
        Attributes:
            badChi2Flag:              1
            badChi2MagneticProbe:     b_nl2_p05,b_nu1_n08,b_nu1_p09
            badChi2MagneticProbeId:   [ 69  75 239]
        ├── DataNode('input')
        │       Dimensions:           (time: 56)
        │       Dimensions without coordinates: time
        │       Data variables:
        │           bVacRadiusProduct  (time) float64 ...
        ├── DataNode('constraints')
        │       Dimensions:  ()
        │       Data variables:
        │           *empty*
        ├── DataNode('pfCircuits')
```

Features 2: Interactive HTML representation



Features 3: Node relationships

- Groups are connected as parent/children (& siblings/ancestors etc...)

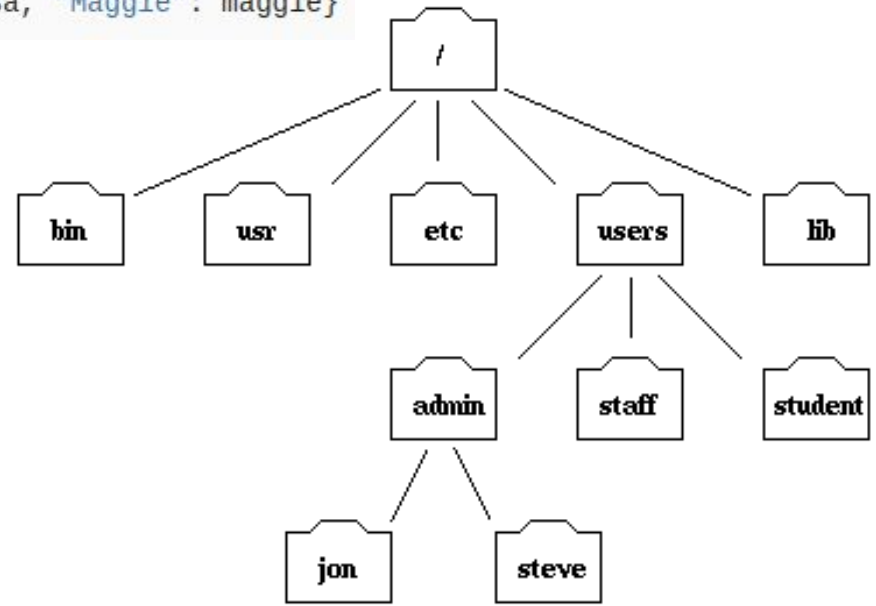
```
homer.children = {"Bart": bart, "Lisa": lisa, "Maggie": maggie}
```

```
DataTree('Abe', parent=None)
├── DataTree('Homer')
│   ├── DataTree('Bart')
│   ├── DataTree('Lisa')
│   └── DataTree('Maggie')
└── DataTree('Herbert')
```

```
In [9]: maggie.parent.name
Out[9]: 'Homer'
```

- Access via filepath-like syntax

```
In [39]: bart.relative_to(lisa)
Out[39]: '../Bart'
```



Part of the filesystem tree

Features 4: Map computations over tree

- Xarray's computation methods are automatically mapped over entire tree below

```
dt.mean(dim="time")
```

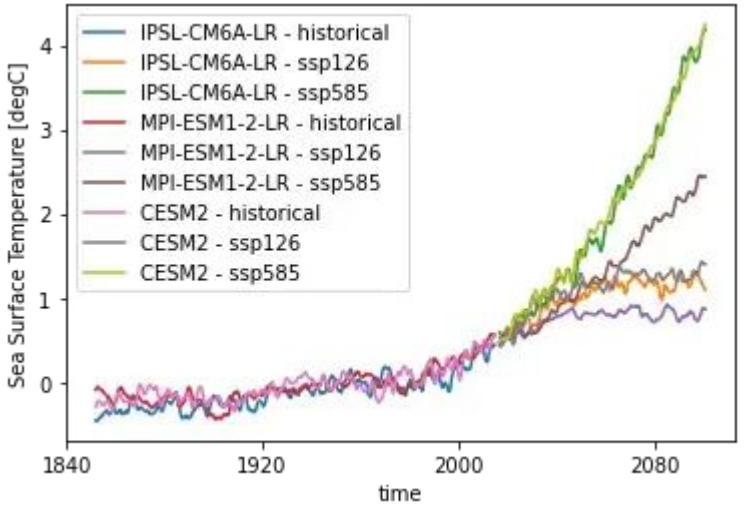
- Can also map custom computation

```
def mean_over_space(ds):
    return ds.mean(dim=["x", "y"])

dt.map_over_subtree(mean_over_space)
```

Example 1: CMIP6 data

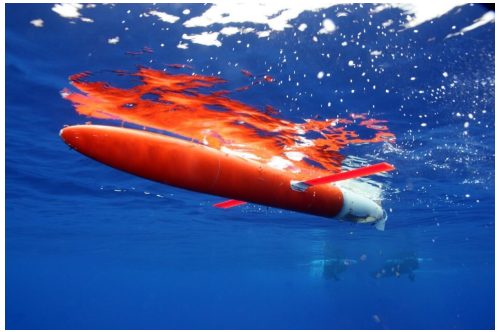
- Julius Busecke and I did a multi-scenario, multi-model analysis
- See pangeo blog post + Julius' talk (3:45 pm tomorrow!)
  - "Reproducible IPCC Science Using Pangeo Tools in the Cloud"




```
DataTree('None', parent=None)
├── DataTree('IPSL-CM6A-LR')
│   ├── DataTree('historical')
│   │       Dimensions:      (y: 332, x: 362, vertex: 4, time: 1980, bnds: 2)
│   │       Coordinates:
│   │         * time         (time) object 1850-01-16 12:00:00 ... 2014-12-16 12:00:00
```

Example 2: Ocean Glider data

- Guilherme Castelao's ocean glider datasets
- Multiple sensors
  - Different time sampling rates



```
├── DataTree('CTD')
│       Dimensions:  (time: 5)
│       Coordinates:
│         * time     (time) datetime64[ns] 2019-12-16T17:22:41 ... 2019-12-16T17:23:13
│       Data variables:
│           press    (time) float64 5.32 4.36 3.4 2.32 1.56
│           temp     (time) float64 15.51 15.51 15.51 15.51 15.51
│           sal      (time) float64 33.56 33.56 33.56 33.56 33.56
│       Attributes:
│           long_name:         CTD sensor
│           type:              CTD
│           type_identifier:   https://vocab.nerc.ac.uk/collection/L05/current/130/
│           maker:             Sea-Bird Scientific
│           maker_identifier:  https://vocab.nerc.ac.uk/collection/L35/current/MAN0013/
│           model:             SBE 41CP CTD
│           model_identifier:  https://vocab.nerc.ac.uk/collection/L22/current/TOOL0669/
│           serial_number:     75
│           calibration_date:  2019-10-22T00:00:00Z
└── DataTree('GPS')
        Dimensions:  (time: 2)
        Coordinates:
          * time     (time) datetime64[ns] 2019-12-16T14:41:00 2019-12-16T17:30:03
        Data variables:
            lat      (time) float64 32.52 32.5
            lon      (time) float64 -119.8 -119.8
        Attributes:
            title:    GPS positions
            comment:  GPS positioning is only possible when the glider is on the surf...
```

Example 3: Multi-resolution maps

- (by Joe Hamman / CarbonPlan)

Future plans

- Integrate datatree upstream into xarray main
- Automatic “tree broadcasting” of operations
- Attribute-like access to child nodes
  - (e.g. `dt.model.<click tab> --> dt.model.experiment_a`)

- **Give me Your Ideas!**

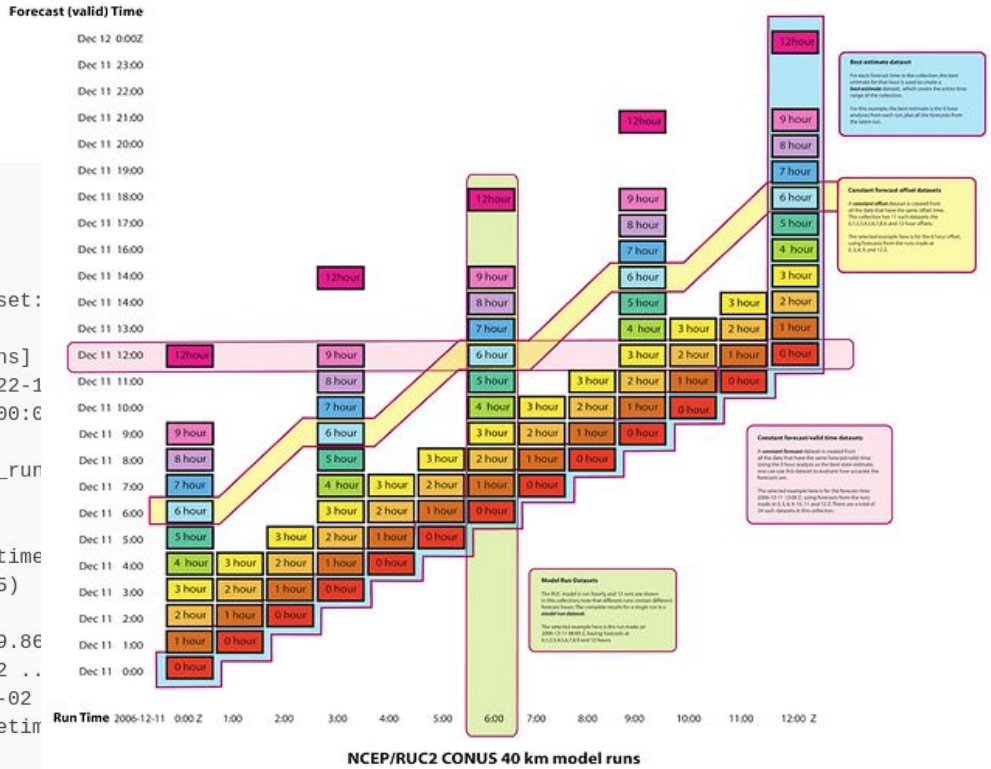
P.S. I am looking for my next big project 😁

github.com/xarray-contrib/datatree

Example 4 (Bonus): Forecast data

- Alex Kerney's Forecast model run collections

```
dt = xarray_fmrc.from_model_runs([ds0, ds1])
dt
DataTree('None', parent=None)
|   Dimensions:                 (forecast_reference_time: 2,
|                                constant_forecast: 242, constant_offset:
|   Coordinates:
|     * forecast_reference_time  (forecast_reference_time) datetime64[ns]
|     * constant_forecast        (constant_forecast) datetime64[ns] 2022-1
|     * constant_offset          (constant_offset) timedelta64[ns] 06:00:0
|   Data variables:
|       model_run_path           (forecast_reference_time) <U29 'model_run
└── DataTree('model_run')
    ├── DataTree('2022-12-01T18:00:00')
    |       Dimensions:                  (forecast_reference_time: 1, time
    |                                     latitude: 220, longitude: 215)
    |       Coordinates:
    |         * longitude                (longitude) float64 -79.95 -79.86
    |         * latitude                 (latitude) float64 27.03 27.12 ..
    |         * time                     (time) datetime64[ns] 2022-12-02
    |         * forecast_reference_time  (forecast_reference_time) datetim
    |       Data variables:
    |           wind_speed               (forecast_reference_time, time, l
    |           wind_from_direction      (forecast_reference_time, time, l
    |       Attributes: (12/178)
```



Example 5 (Bonus): xRadar

- We facilitate `datatree.DataTree` to bundle the different sweeps of a radar volume into one structure. These sweep datasets are essentially `xarray.Dataset` which contain metadata attributes and variables (`xarray.DataArray`).