Pangeo
A community-driven effort for Big Data geoscience
What Drives Progress in Oceanography?

New Ideas

\[ E = \frac{\rho_0 |\mathbf{U}|}{\pi} \int_{|\mathbf{U}|}^{N|\mathbf{U}|} P_{1D}(k) \sqrt{N^2 - |\mathbf{U}|^2} k^2 \sqrt{|\mathbf{U}|^2 k^2} \, dk \]

New Observations

New Simulations
Major Science Questions

• How is energy transferred across scales and dissipated in the ocean?

• How do mesoscales / submesoscales / tides / internal waves contribute to the transport of heat / salt / dissolved tracers vertically and horizontally?

• How does abyssal flow navigate complex small-scale topography (e.g. shelf overflows, Indonesian Throughflow, abyssal canyons)?

• How should we represent these processes in coarse resolution climate models?

dozens of high impact papers are waiting to be written!
My Big Data Journey

- 2013: started at Columbia
- 2014: wandered the desert
- 2015: discovered Big Data
My Big Data Journey

- 2013: discovered Big Data
- 2014: started at Columbia
- 2015: wandered the desert
- 2016: discovered xarray!
Scientific Python for Data Science

Growth of major programming languages
Based on Stack Overflow question views in World Bank high-income countries

Stack Overflow Traffic to Questions About Selected Python Packages
Based on visits to Stack Overflow questions from World Bank high-income countries

source: stackoverflow.com
Scientific Python for Data Science

Iris  
cartopy  
GCM  
aospy  
TensorFlow  
DASK  
pandas  
xarray  
matplotlib  
SciPy  
NumPy  
IPython  
Python  
Jupyter

Credit: Stephan Hoyer, Jake Vanderplas (SciPy 2015)
Xarray Dataset: Multidimensional Variables with coordinates and metadata

Data variables used for computation

Coordinates describe data

Indexes align data

 Attributes metadata ignored by operations

“netCDF meets pandas.DataFrame”

Credit: Stephan Hoyer
import xarray as xr

ds = xr.open_dataset('NOAA_NCDC_ERSST_v3b_SST.nc')

ds

<xarray.Dataset>
Dimensions: (lat: 89, lon: 180, time: 684)
Coordinates:
  * lat    (lat) float32 -88.0 -86.0 -84.0 -82.0 -80.0 -78.0 -76.0 -74.0 ...
  * lon    (lon) float32 0.0 2.0 4.0 6.0 8.0 10.0 12.0 14.0 16.0 18.0 20.0 ...
  * time   (time) datetime64[ns] 1960-01-15 1960-02-15 1960-03-15 ...
Data variables:
  sst     (time, lat, lon) float64 nan nan nan nan nan nan nan nan ...
Attributes:
  Conventions: IRIDL
  source: https://iridl.ldeo.columbia.edu/SOURCES/.NOAA/.NCDC/.ERSST/...
xarray: label-based selection

# select and plot data from my birthday
ds.sst.sel(time='1982-08-07', method='nearest').plot()
# zonal and time mean temperature
ds.sst.mean(dim=('time', 'lon')).plot()
xarray: grouping and aggregation

```python
sst_clim = sst.groupby('time.month').mean(dim='time')
sst_anom = sst.groupby('time.month') - sst_clim
nino34_index = (sst_anom.sel(lat=slice(-5, 5), lon=slice(190, 240))
                 .mean(dim=('lon', 'lat'))
                 .rolling(time=3).mean(dim='time'))
nino34_index.plot()
```
xarray
https://github.com/pydata/xarray

• label-based indexing and arithmetic

• interoperability with the core scientific Python packages (e.g., pandas, NumPy, Matplotlib)

• out-of-core computation on datasets that don’t fit into memory (thanks dask!)

• wide range of input/output (I/O) options: netCDF, HDF, geoTIFF, zarr

• advanced multi-dimensional data manipulation tools such as group-by and resampling
Legacy software

NASA Panoply

NCL
NCAR Command Language

INGRID

PANGEA
Complex computations represented as a graph of individual tasks.

Scheduler optimizes execution of graph.
Example Calculation: Take the Mean!

multidimensional array

<table>
<thead>
<tr>
<th>8</th>
<th>8</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x', 0, 0)</td>
<td>(x', 0, 1)</td>
<td>(x', 0, 2)</td>
</tr>
<tr>
<td>(x', 1, 0)</td>
<td>(x', 1, 1)</td>
<td>(x', 1, 2)</td>
</tr>
<tr>
<td>(x', 2, 0)</td>
<td>(x', 2, 1)</td>
<td>(x', 2, 2)</td>
</tr>
<tr>
<td>(x', 3, 0)</td>
<td>(x', 3, 1)</td>
<td>(x', 3, 2)</td>
</tr>
</tbody>
</table>

serial execution (a loop)

read chunk from disk → reduce → read chunk from disk → reduce → read chunk from disk → reduce

store → store → store → reduce
Example Calculation: Take the Mean!

Multidimensional array

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>(x', 0, 0)</td>
<td>(x', 0, 1)</td>
</tr>
<tr>
<td>5</td>
<td>(x', 1, 0)</td>
<td>(x', 1, 1)</td>
</tr>
<tr>
<td>5</td>
<td>(x', 2, 0)</td>
<td>(x', 2, 1)</td>
</tr>
<tr>
<td>5</td>
<td>(x', 3, 0)</td>
<td>(x', 3, 1)</td>
</tr>
</tbody>
</table>

Parallel execution (dask graph)
My Big Data Journey

- 2013: started at Columbia
- 2014: discovered Big Data
- 2015: wandered the desert
- 2016: discovered xarray!
- 2017: first Pangeo workshop
- 2018: used xarray on datasets up to ~200 GB

connected with xarray community
Pangeo Project goals

• Foster collaboration around the open source scientific python ecosystem for ocean / atmosphere / land / climate science.

• Support the development with domain-specific geoscience packages.

• Improve scalability of these tools to handle petabyte-scale datasets on HPC and cloud platforms.
My Big Data Journey

- 2013: started at Columbia
- 2014: discovered Big Data
- 2015: wandered the desert
- 2016: discovered xarray!
- 2017: used xarray on datasets up to ~200 GB
- 2018: connected with fantastic xarray community

Earthcube proposal awarded
Earthcube Award Team

Lamont-Doherty Earth Observatory
Columbia University | Earth Institute

Ryan Abernathey, Chiara Lepore, Michael Tippet, Naomi Henderson, Richard Seager

National Center for Atmospheric Research

Kevin Paul, Joe Hamman, Ryan May, Davide Del Vento

Anaconda

Matthew Rocklin
Other Contributors

Jacob Tomlinson, Niall Roberts, Alberto Arribas
Developing and operating Pangeo environment to support analysis of UK Met office products

Rich Signell
Deploying Pangeo on AWS to support analysis of coastal ocean modeling

Justin Simcock
Operating Pangeo in the cloud to support Climate Impact Lab research and analysis
Supporting Pangeo via SWOT mission and recently funded ACCESS award to UW / NCAR

Yuvi Panda, Chris Holdgraf
Spending lots of time helping us make things work on the cloud
**Pangeo Architecture**

“Analysis Ready Data” stored on globally-available distributed storage.

Parallel computing system allows users deploy clusters of compute nodes for data processing.

Dask tells the nodes what to do.

Xarray provides data structures and intuitive interface for interacting with datasets.

Jupyter for interactive access remote systems.

Cloud / HPC

Distributed storage

web browser

end user
## Build your own pangeo

<table>
<thead>
<tr>
<th>Storage Formats</th>
<th>HDF</th>
<th>OPeNDAP</th>
<th>Cloud Optimized COG/Zarr/Parquet/etc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ND-Arrays</td>
<td>NumPy</td>
<td>DASK</td>
<td>More coming…</td>
</tr>
<tr>
<td>Data Models</td>
<td>xarray</td>
<td>Iris</td>
<td>pandas</td>
</tr>
<tr>
<td>Processing Mode</td>
<td>Jupyter</td>
<td>Interactive</td>
<td>Batch</td>
</tr>
<tr>
<td>Compute Platform</td>
<td>HPC</td>
<td>Cloud</td>
<td>Local</td>
</tr>
</tbody>
</table>
Pangeo Deployments

pangeo.pydata.org

Over 1000 unique users since March

http://pangeo.io/deployments.html
Climate Data in the Cloud ERA

Traditional Approach: A Data Access Portal

Data Granules (netCDF files)

file.0001.nc
file.0002.nc
file.0003.nc
file.0004.nc

Data Access Server

Client
Client
Client

Data Center

Internet
Climate Data in the Cloud ERA

Direct Access to Cloud Object Storage

Data Granules
(netCDF files or something new)

Cloud Object Storage

chunk.0.0.0
chunk.0.0.1
chunk.0.0.2
chunk.0.0.3

Cloud Compute
Instances

Client
Client
Client

Catalog

Cloud Data Center
File/ Block storage

- Operating system provides mechanism to read / write files and directories (e.g. POSIX).

- Seeking and random access to bytes within files is fast.

- “Most file systems are based on a block device, which is a level of abstraction for the hardware responsible for storing and retrieving specified blocks of data”

Image credit: https://blog.ubuntu.com/2015/05/18/what-are-the-different-types-of-storage-block-object-and-file
Object storage

- An object is a collection of bytes associated with a unique identifier
- Bytes are read and written with http calls
- Significant overhead each individual operation
- Application level (not OS dependent)
- Implemented by S3, GCS, Azure, Ceph, etc.

Image credit: https://blog.ubuntu.com/2015/05/18/what-are-the-different-types-of-storage-block-object-and-file
Python library for storage of chunked, compressed ND-arrays

Developed by Alistair Miles (Imperial) for genomics research (@alimanfoo)

Arrays are split into user-defined chunks; each chunk is optional compressed (zlib, zstd, etc.)

Can store arrays in memory, directories, zip files, or any python mutable mapping interface (dictionary)

External libraries (s3fs, gcsf) provide a way to store directly into cloud object storage
**zarr**

Example .zarray file (json)

```
{
    "chunks": [5, 720, 1440],
    "compressor": {
        "blocksize": 0,
        "clevel": 3,
        "cname": "zstd",
        "id": "blosc",
        "shuffle": 2
    },
    "dtype": "<f8",
    "fill_value": "NaN",
    "filters": null,
    "order": "C",
    "shape": [8901, 720, 1440],
    "zarr_format": 2
}
```
Example .attrs file (json)

```json
{
  "_ARRAY_DIMENSIONS": [
    "time",
    "latitude",
    "longitude"
  ],
  "comment": "The sea level anomaly is the sea surface height above mean sea surface; it is referenced to the [1993, 2012] period; see the product user manual for details",
  "coordinates": "crs",
  "grid_mapping": "crs",
  "long_name": "Sea level anomaly",
  "standard_name": "sea_surface_height_above_sea_level",
  "units": "m"
}
```
Developed new xarray backend which allows xarray to read and write directly to a Zarr store (with @jhamman)

It was pretty easy! Data models are quite similar

Automatic mapping between zarr chunks $\leftarrow \rightarrow$ dask chunks

We needed to add a custom, “hidden” attribute (_ARRAY_DIMENSIONS) to give the zarr arrays dimensions
Preparing datasets for zarr cloud storage

1. Open the original data files into a single `xarray` dataset with reasonable chunks

   ```python
   ds = xr.open_mfdataset('bunch_o_files_*.nc', chunks={'time': 1})
   ```

2. Export to zarr

   ```python
   ds.to_zarr('/path/to/zarr/directory')
   --- or ---
   ds.to_zarr(gcsamp_object)
   ```

3. [maybe] upload to cloud storage

   ```bash
   $ gcsutil -m cp -r /path/to/zarr/directory gs://pangeo-data/path
   ```
Where is Pangeo Going?

- Pangeo + Binder! https://github.com/pangeo-data/pangeo-binder

- Custom JupyterLab extensions (dask dashboards, cluster monitoring, data catalog browsing)

- User management (home directories, scratch space, etc.)

- Domain-specific cloud environments: ocean.pangeo.io, atmos.pangeo.io, astro.pangeo.io [?]
Contribute to xarray, dask, zarr, jupyterhub, etc.

Access an existing Pangeo deployment on an HPC cluster, or cloud resources (eg. pangeo.pydata.org)

Adapt Pangeo elements to meet your projects needs (data portals, etc.) and give feedback via github: github.com/pangeo-data/pangeo

http://pangeo.io