Dynamic visualisation in the IPython Notebook

www.cawcr.gov.au

Brianna Laugher
Hobart, Tasmania
July 2013, PyCon AU

Australian Government
Bureau of Meteorology

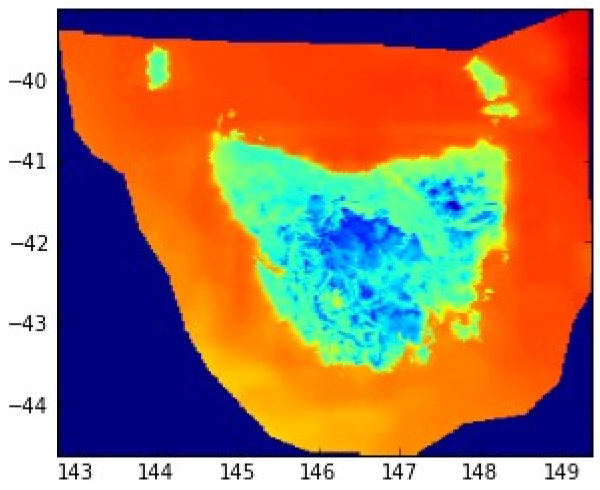
The Centre for Australian Weather and Climate Research
A partnership between CSIRO and the Bureau of Meteorology

CSIRO

What we have...

```
from pydap.client import open_url
dataset = open_url("http://127.0.0.1:8001/IDT71003_TAS_MinT_SFC.nc")

mint = dataset['MinT_SFC']
lats = dataset['latitude']
lons = dataset['longitude']

# Just pull out data for the first day
day0 = mint[0]
data = np.squeeze(day0)
```

```
extent = [min(lons), max(lons), min(lats), max(lats)]
_ = plt.imshow(data, origin='lower', vmin=0, vmax=20, extent=extent)
```

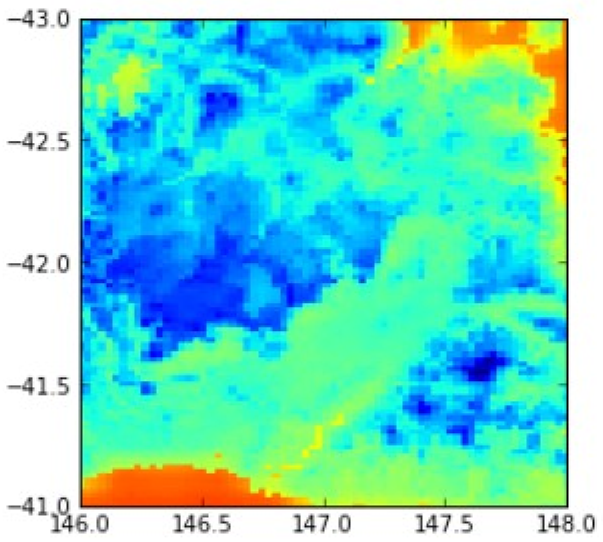


Australian Government
Bureau of Meteorology

The Centre for Australian

What we have...

```
# Plot around Hobart.
# Setting interpolation to nearest means NO interpolation
# ie. we see the data as blocky because it actually is.
_ = plt.imshow(data, interpolation='nearest', origin='lower',
               vmin=0, vmax=20, extent=extent)
_ = plt.axis([146, 148, -41, -43])
```



The problem

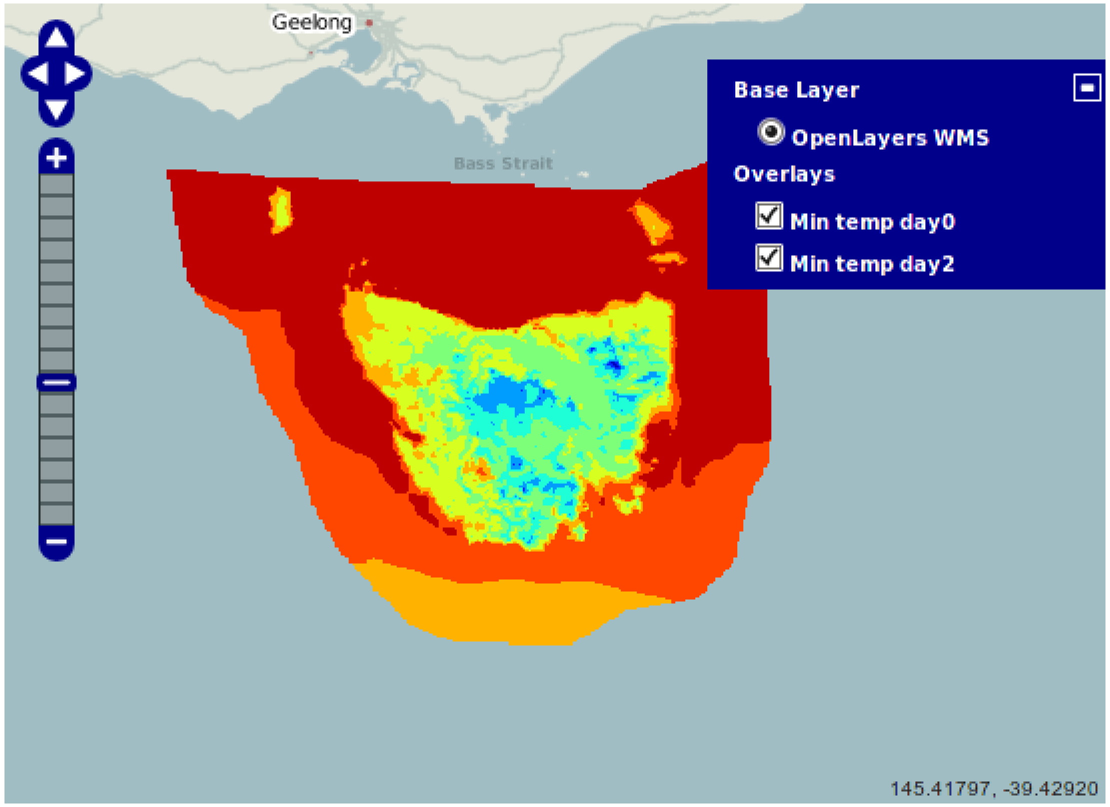
Can't zoom
Can't pan
Can't layer
Can't sample by clicking

IPython Notebook makes it easy to explore datasets and plot them with matplotlib...

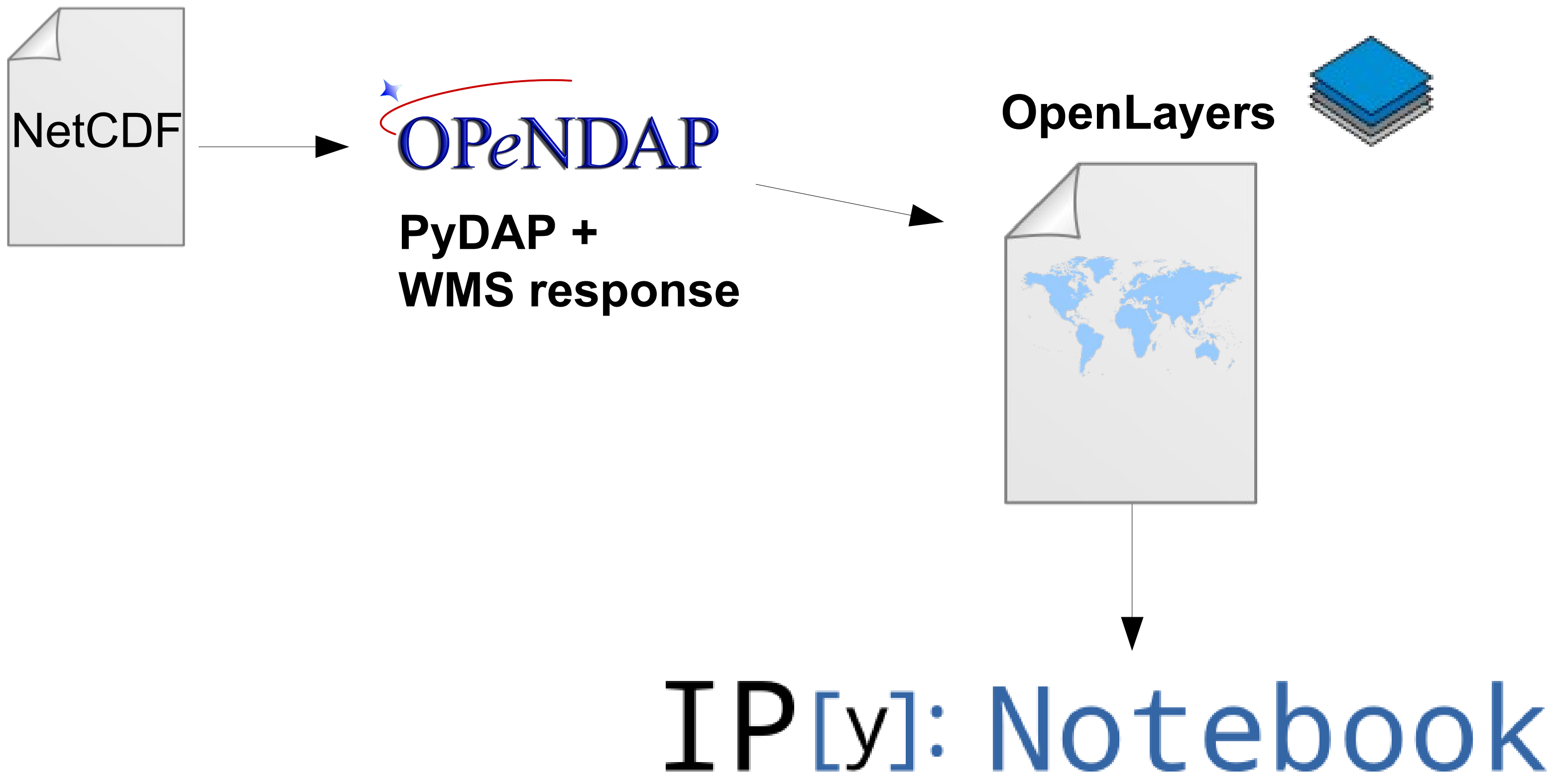
...but for gridded datasets, it really would be nice to have them on a map!

...what we want

```
import dapbook
a = dapbook.PydapWMS(server='http://127.0.0.1:8001/IDT71003_TAS_MinT_SFC.nc.wms',
                     layers=[('MinT_SFC', 'Min temp day0', 0),
                             ('MinT_SFC', 'Min temp day2', 2),
                     ], centerlat=-42, centerlon=147, initialzoom=6,
                     mapwidth=550, mapheight=400)
a.html
```



How to get there?



IPython notebook

“a web-based interactive computational environment where you can combine code execution, text, mathematics, plots and rich media into a single document”

- Inspired by Mathematica, SAGE notebooks
- Built-in support for numpy, matplotlib
- Half-interpreter, half-script
- Great way to “show your work”
- Useful for tuning fiddly APIs (ahem matplotlib)
- Perfect for tutorials!

Run a local server, and/or

Publish your notebook as .ipynb and use http://nbviewer.ipython.org/

IPython notebook

```
pip install ipython
pip install tornado pyzmq  # needed for notebook, not ipython shell
ipython notebook --pylab inline
```

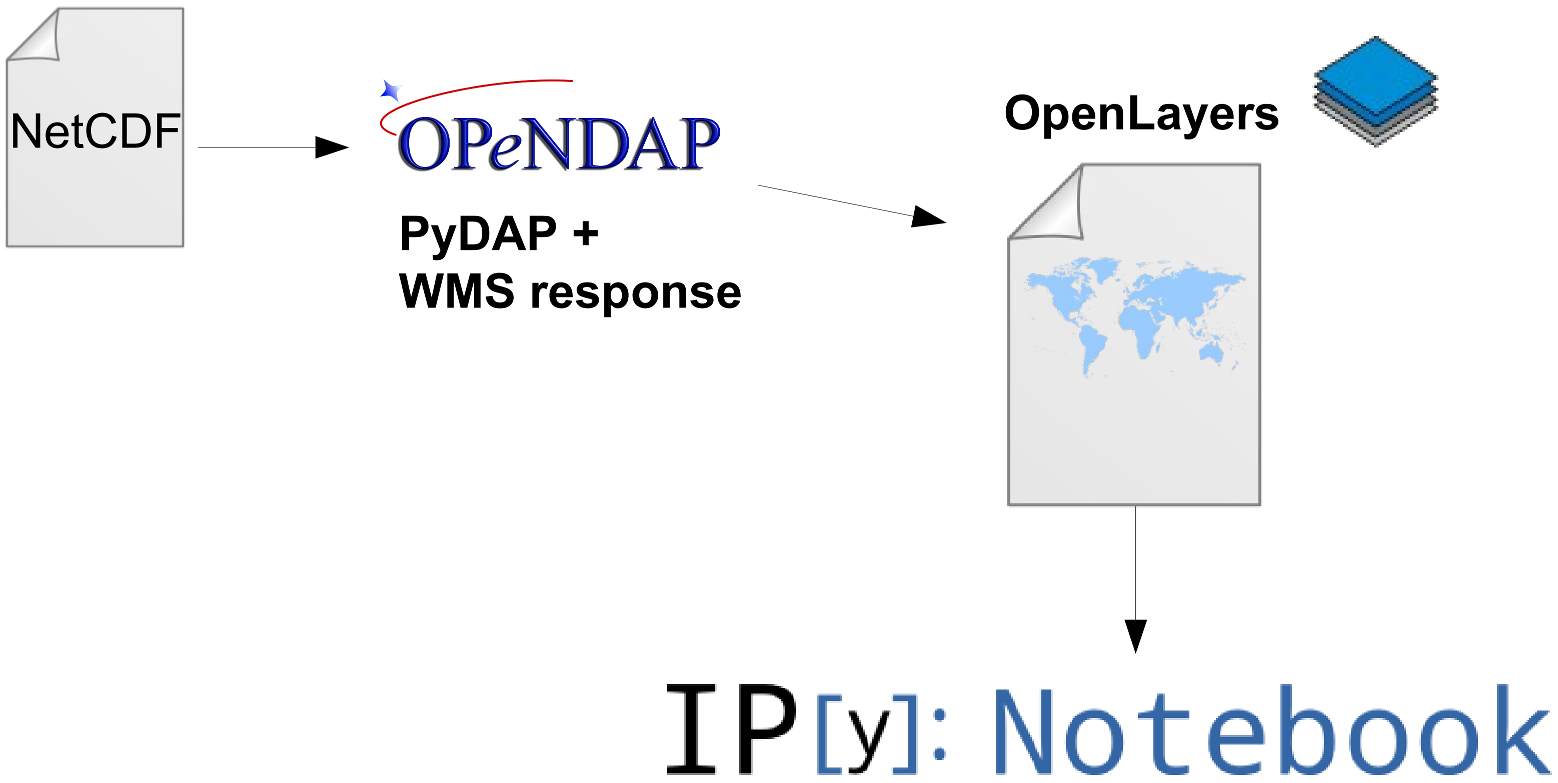
IPython notebook

Display system – define "rich reprs" for HTML, JSON, PNG, JPEG, SVG, LaTeX

```
from IPython.display import Image, HTML
```

If you define a class with 'png', 'svg' properties that return instances of IPython.display.Image – hey presto, you are notebook friendly!

How to get there?



JavaScript mapping libraries

Leaflet and OpenLayers:

- **Specify layers as “tile layers” or “WMS tile layers” (different APIs)**
- **Can set projection as required**
- **Can reproject points and vectors (but not map tiles) on-the-fly**

WMS

Australian Government
Bureau of Meteorology

The Centre for Australian

Web Map Service

Supports 2+ request types – GetMap, GetCapabilities

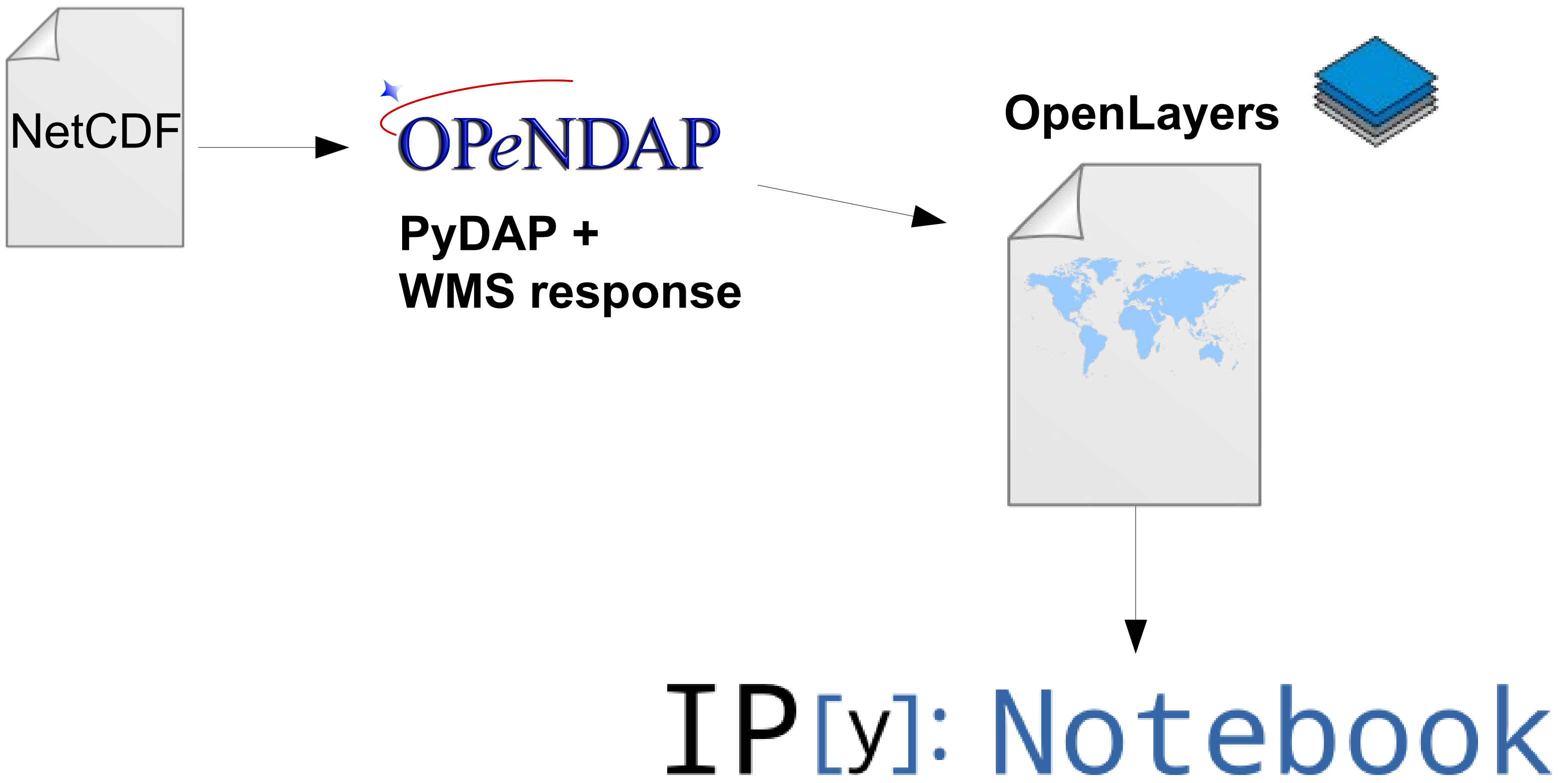
http://test.pydap.org/coads.nc.wms?SST[0]&LAYERS=SST&TRANSPARENT=true&FORMAT=image%2Fpng&SERVICE=WMS&VERSION=1.1.1&REQUEST=GetMap&STYLES=&SRS=EPSG%3A900913&BBOX=-180,-90,1252164.27125,1252254.27125&WIDTH=256&HEIGHT=256

REQUEST=GetMap
SERVICE=WMS
VERSION=1.1.1

LAYERS=MinT_SFC
SRS=EPSG:900913
BBOX=-180,-10018844.17,10018574.17,-90

FORMAT=image/png
TRANSPARENT=true
HEIGHT=256
WIDTH=256

How to get there?



PyDAP

2 parts: OPeNDAP server, OPeNDAP client

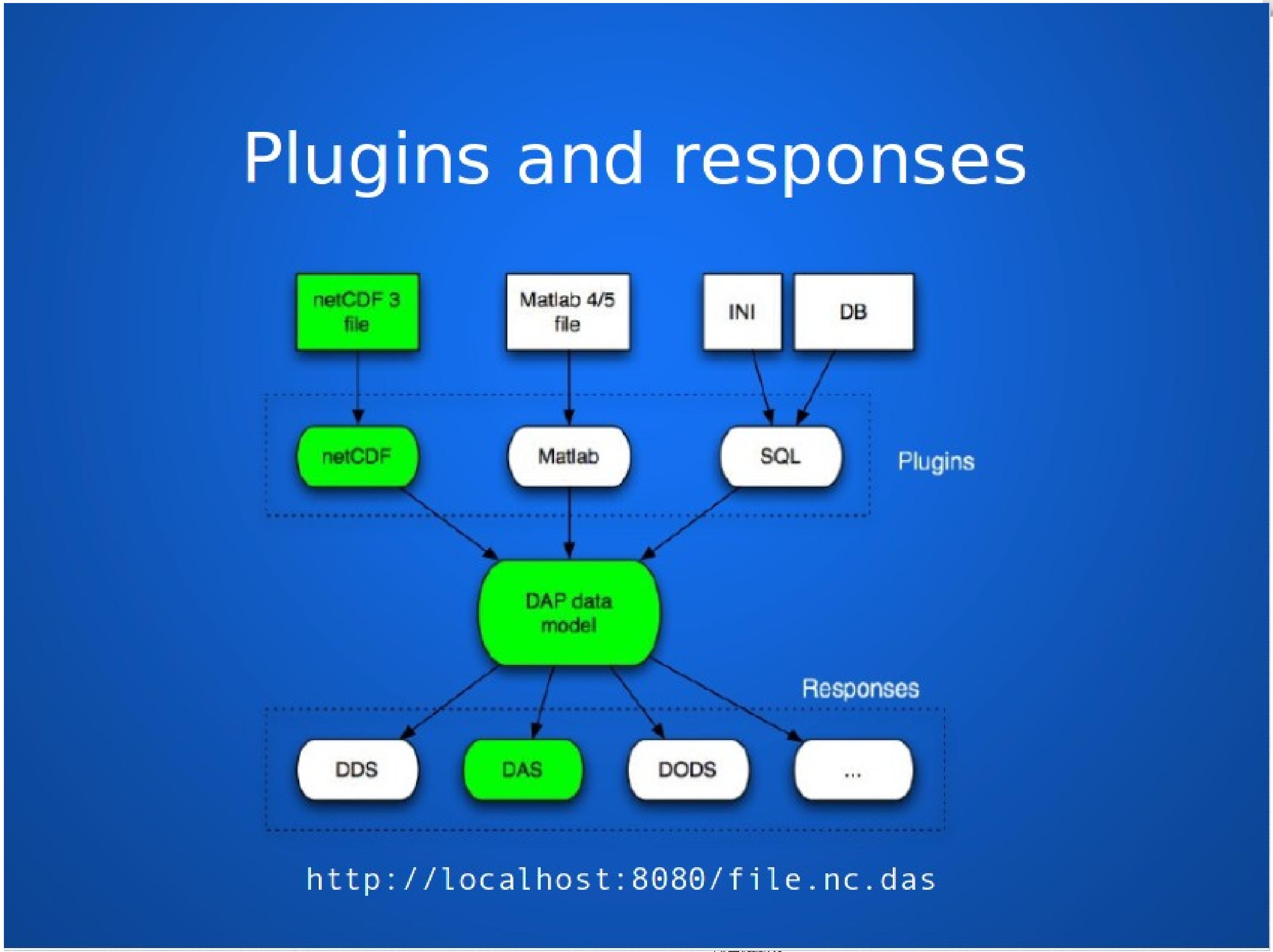
Server

- WSGI app
- Handlers (input): NetCDF, HDF5, SQL, CSV, remote!, ...
- Responses (ouput): DAS/DDS/DODS, HTML, ASCII
  WMS, KML, XLS, ...

Client

- “Lazy loading” of data from any OPeNDAP server into numpy (incl. slices, subsets)

PyDAP

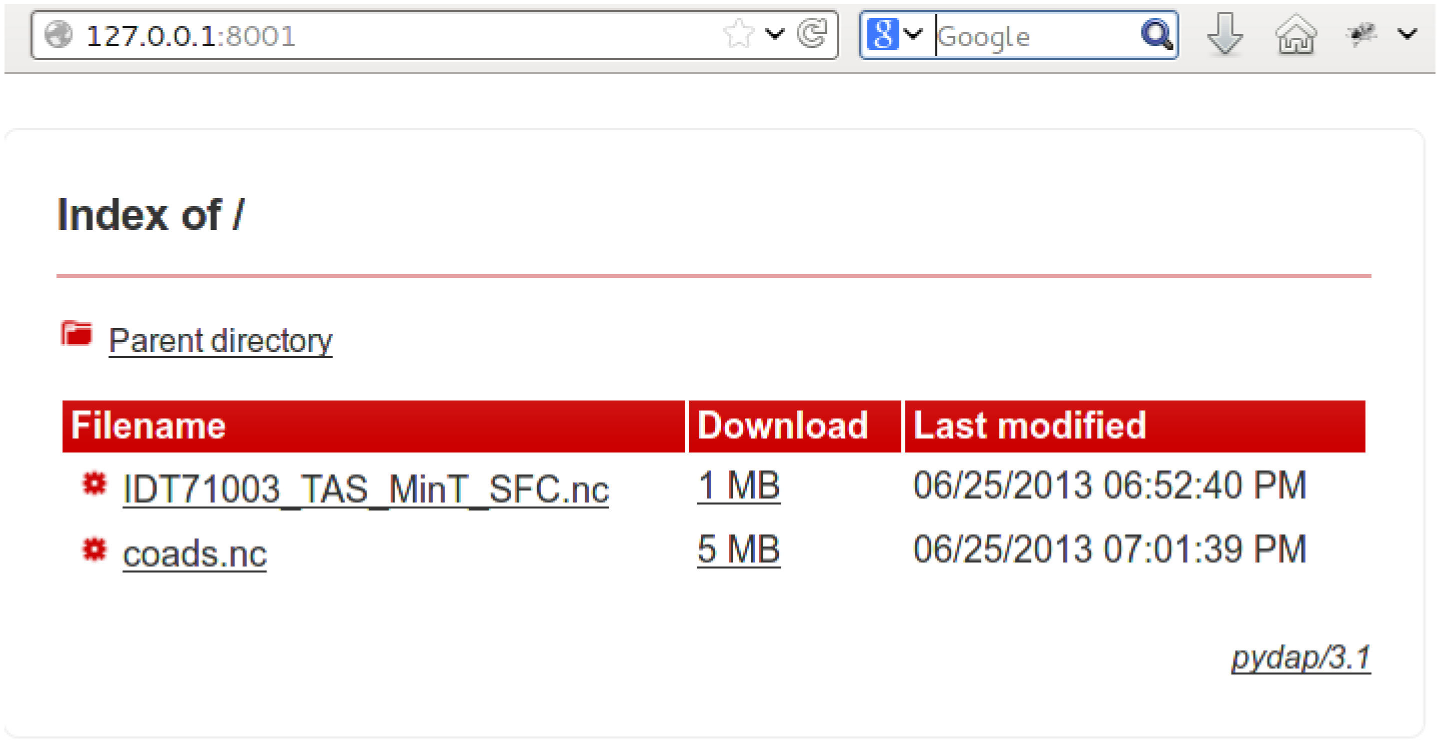

Plugins and responses

http://localhost:8080/file.nc.das

PyDAP

```
pip install pydap pydap.handlers.netcdf pydap.responses.wms
paster create -t pydap myserver
cp mydata.nc myserver/data/

# starts a server like http://test.pydap.org/
# run server with 4 workers, better for serving up map tiles
gunicorn_paster -w 4 -b 127.0.0.1:8001 myserver/server.ini
```

PyDAP



127.0.0.1:8001

Google

Index of /

Parent directory

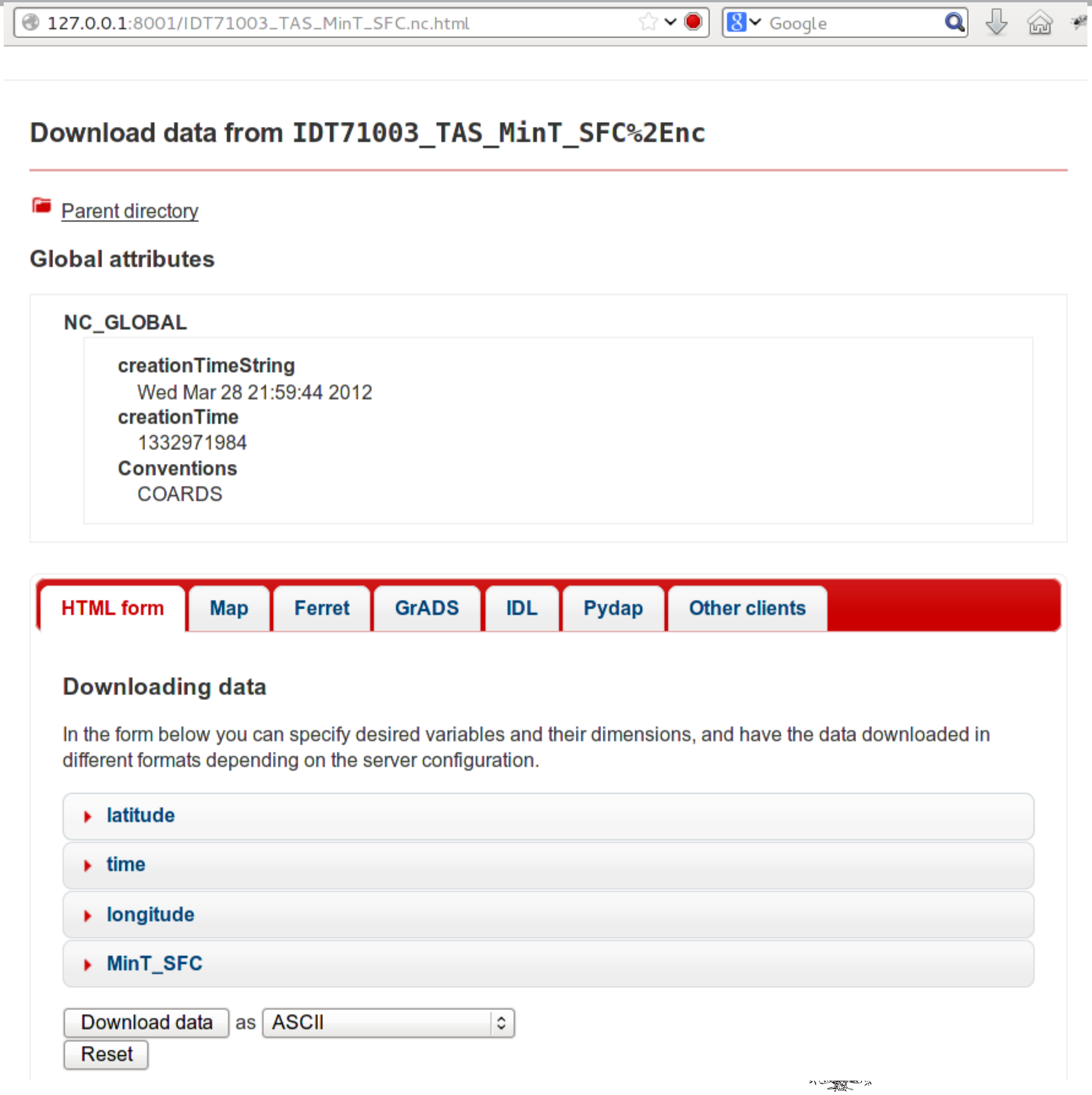
Filename	Download	Last modified
IDT71003_TAS_MinT_SFC.nc	1 MB	06/25/2013 06:52:40 PM
coads.nc	5 MB	06/25/2013 07:01:39 PM

pydap/3.1

Australian Government
Bureau of Meteorology

The Centre for Australian

PyDAP



127.0.0.1:8001/IDT71003_TAS_MinT_SFC.nc.html

Google

Download data from IDT71003_TAS_MinT_SFC%2Enc

Parent directory

Global attributes

NC_GLOBAL

creationTimeString
Wed Mar 28 21:59:44 2012
creationTime
1332971984
Conventions
COARDS

HTML form | Map | Ferret | GrADS | IDL | Pydap | Other clients

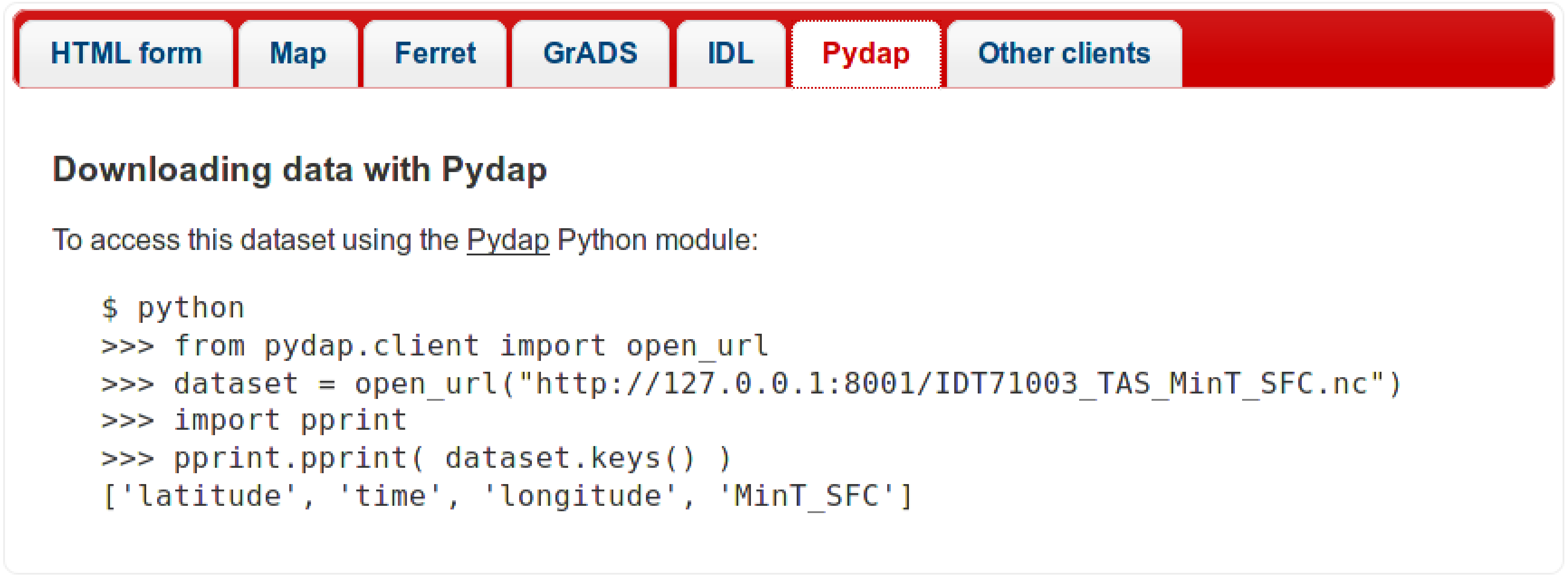
Downloading data

In the form below you can specify desired variables and their dimensions, and have the data downloaded in different formats depending on the server configuration.

- latitude
- time
- longitude
- MinT_SFC

Download data as ASCII
Reset

PyDAP



Downloading data with Pydap

To access this dataset using the Pydap Python module:

```
$ python
>>> from pydap.client import open_url
>>> dataset = open_url("http://127.0.0.1:8001/IDT71003_TAS_MinT_SFC.nc")
>>> import pprint
>>> pprint.pprint( dataset.keys() )
['latitude', 'time', 'longitude', 'MinT_SFC']
```

Projections 101

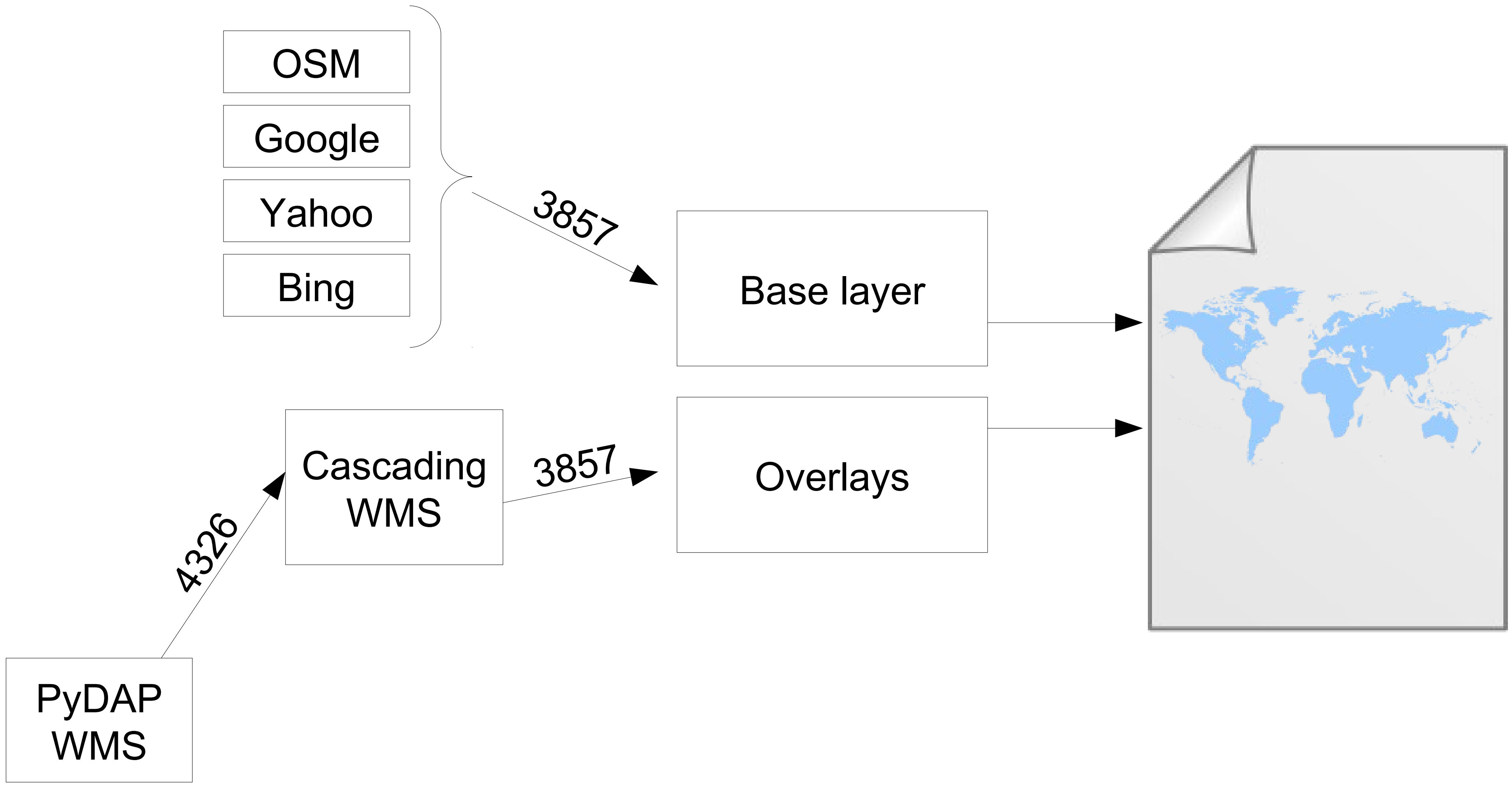
Mapping libraries can reproject points and vectors, but not map tiles...

Base layers and overlays need to be requested in the same projection!

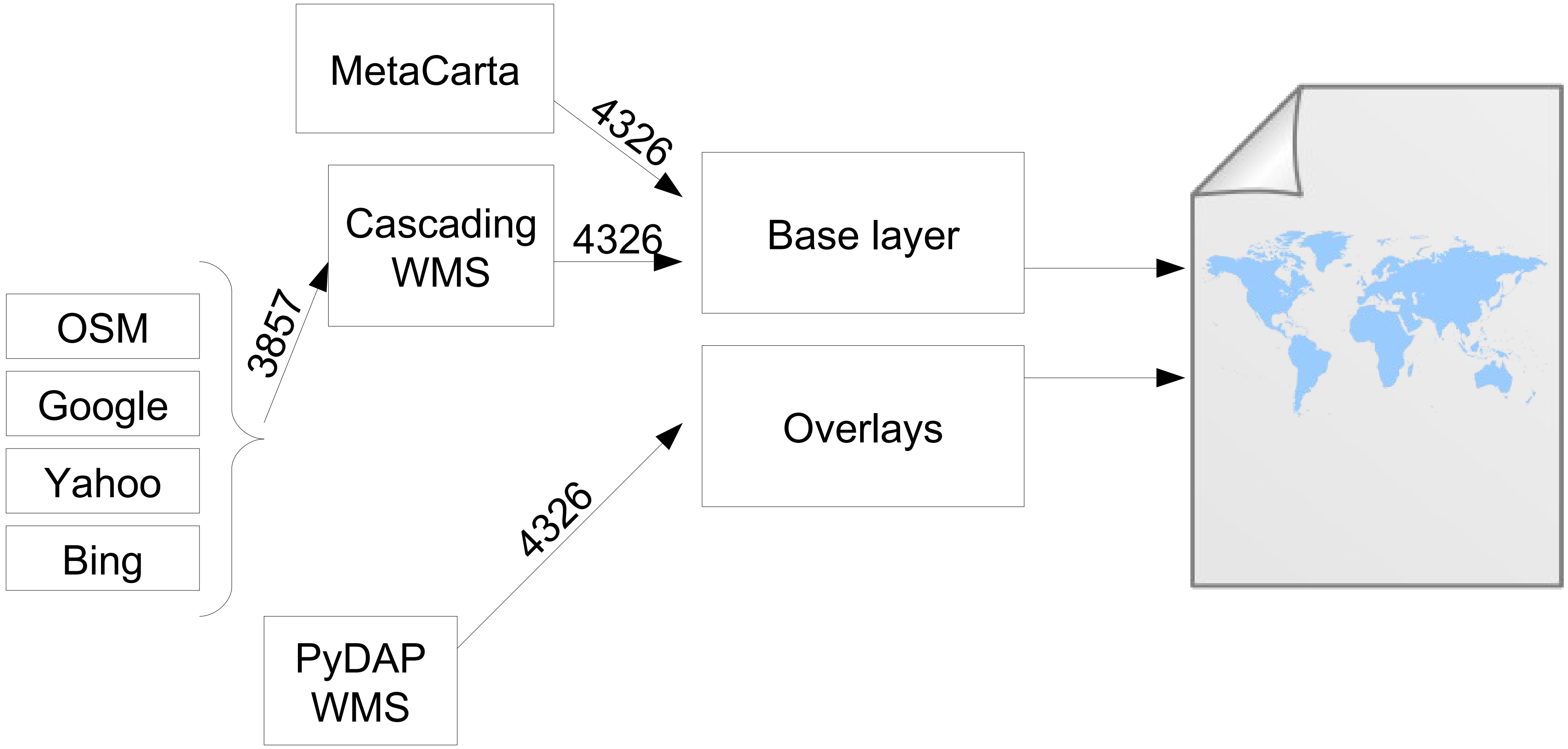
Projections 101

CRS	**EPSG:4326**	**EPSG:3857 (aka EPSG:900913)**
Example of	Geographic coordinate system (represents a globe/part of a globe)	Projected coordinate system, “Spherical Mercator” (represents a map)
Used by	“GIS enthusiasts”, Metacarta	“almost all free and commercial tile providers” (Google, OSM, Yahoo, Bing)
Coords as	Lat-lons in decimal degrees	“Metres”
PyDAP WMS can serve map tiles in	√	
OpenLayers supports by default	√	√
Leaflet supports by default	√ But... issue #1207 “EPSG 4326 Support Broken for TileLayers” and...CloudMade doesn't provide WMS.	√

Option 1: Render overlays in EPSG:3857



Option 2: Pick base layer in EPSG:4326



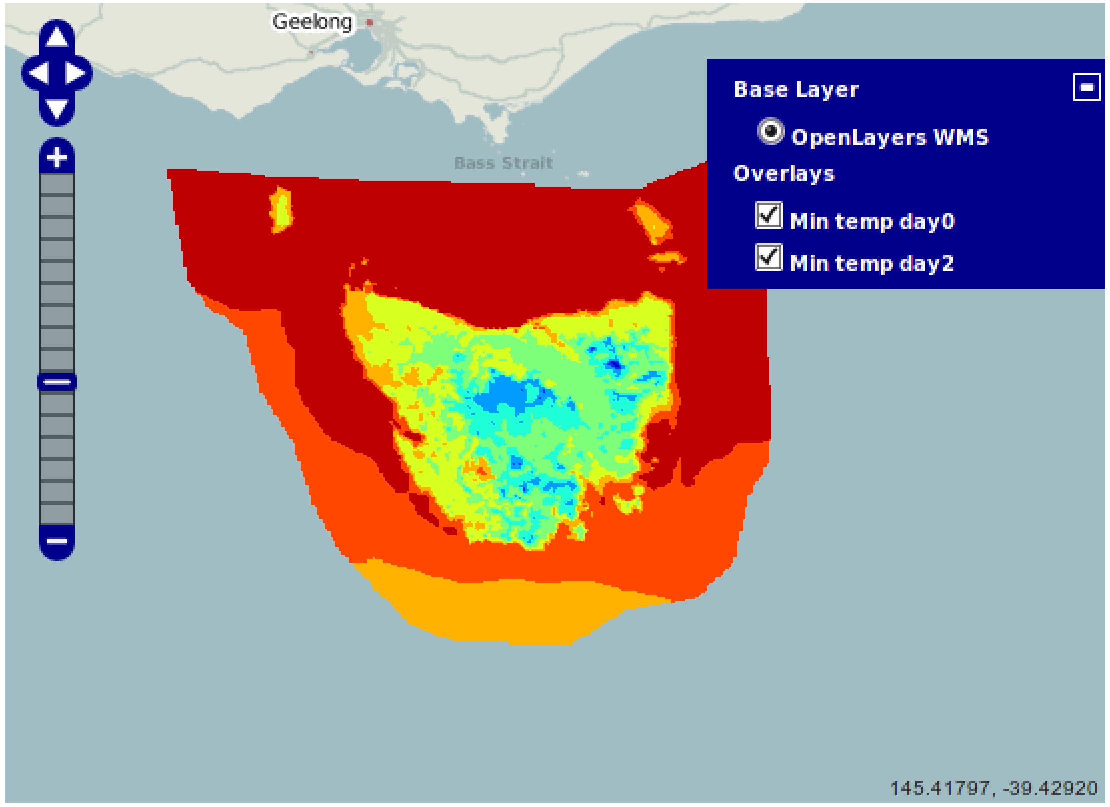
Cascading WMS

- MapServer
  - C
  - MIT license
- GeoServer + GeoWebCache
  - Java
  - GPL, LGPL licenses
  - http://maps.opengeo.org/geowebcache/demo
    - Includes OpenStreetMap, NASA “Blue Marble”

```python
import os
from genshi.template import TemplateLoader
from IPython.display import HTML

loader = TemplateLoader(os.getcwd())


class PydapWMS(object):
    """A simple object for displaying data from a Pydap WMS."""
    def __init__(self, server='http://test.pydap.org/coads.nc.wms',
                 layers=None, centerlat=0, centerlon=0, initialzoom=2,
                 mapwidth=400, mapheight=300, template='template.html'):
        self.wmsserverurl = server
        self.layers = layers if layers else [('SST', 'Sea Surface Temperature', 0)]
        self.centerlat = centerlat
        self.centerlon = centerlon
        self.initialzoom = initialzoom
        self.mapwidth = mapwidth
        self.mapheight = mapheight
        self.template = template

    def __repr__(self):
        return 'PydapWMS({})'.format(str(self.__dict__))

    @property
    def html(self):
        tmpl = loader.load(self.template)
        html = tmpl.generate(**self.__dict__).render('html', doctype='html')
        src = 'data:text/html;base64,{}'.format(html.encode('base64'))
        iframe = '<iframe src="{}" width="{}" height="{}"></iframe>'
        return HTML(iframe.format(src, self.mapwidth + 50, self.mapheight + 50))
```

Back to the notebook

```
import dapbook
a = dapbook.PydapWMS(server='http://127.0.0.1:8001/IDT71003_TAS_MinT_SFC.nc.wms',
                     layers=[('MinT_SFC', 'Min temp day0', 0),
                             ('MinT_SFC', 'Min temp day2', 2),
                     ], centerlat=-42, centerlon=147, initialzoom=6,
                     mapwidth=550, mapheight=400)
a.html
```



Base Layer
- ● OpenLayers WMS

Overlays
- ✓ Min temp day0
- ✓ Min temp day2

Credits

With thanks to:

Nathan Faggian

Roberto de Almeida

James Sofra

Danielle Madeley

Roald de Wit

Credits

Map tiles photo: “Carcassonne” by Tom & Katrien, licensed CC-BY-SA.
http://www.flickr.com/photos/inferis/283379928/

Australian Government
Bureau of Meteorology

The Centre for Australian Weather and Climate Research
A partnership between CSIRO and the Bureau of Meteorology

CSIRO

Brianna Laugher

Email: b.laugher@bom.gov.au
Web: www.cawcr.gov.au

Thank you

www.cawcr.gov.au

TODOs

Get sample at point (pydap.responses.json?)

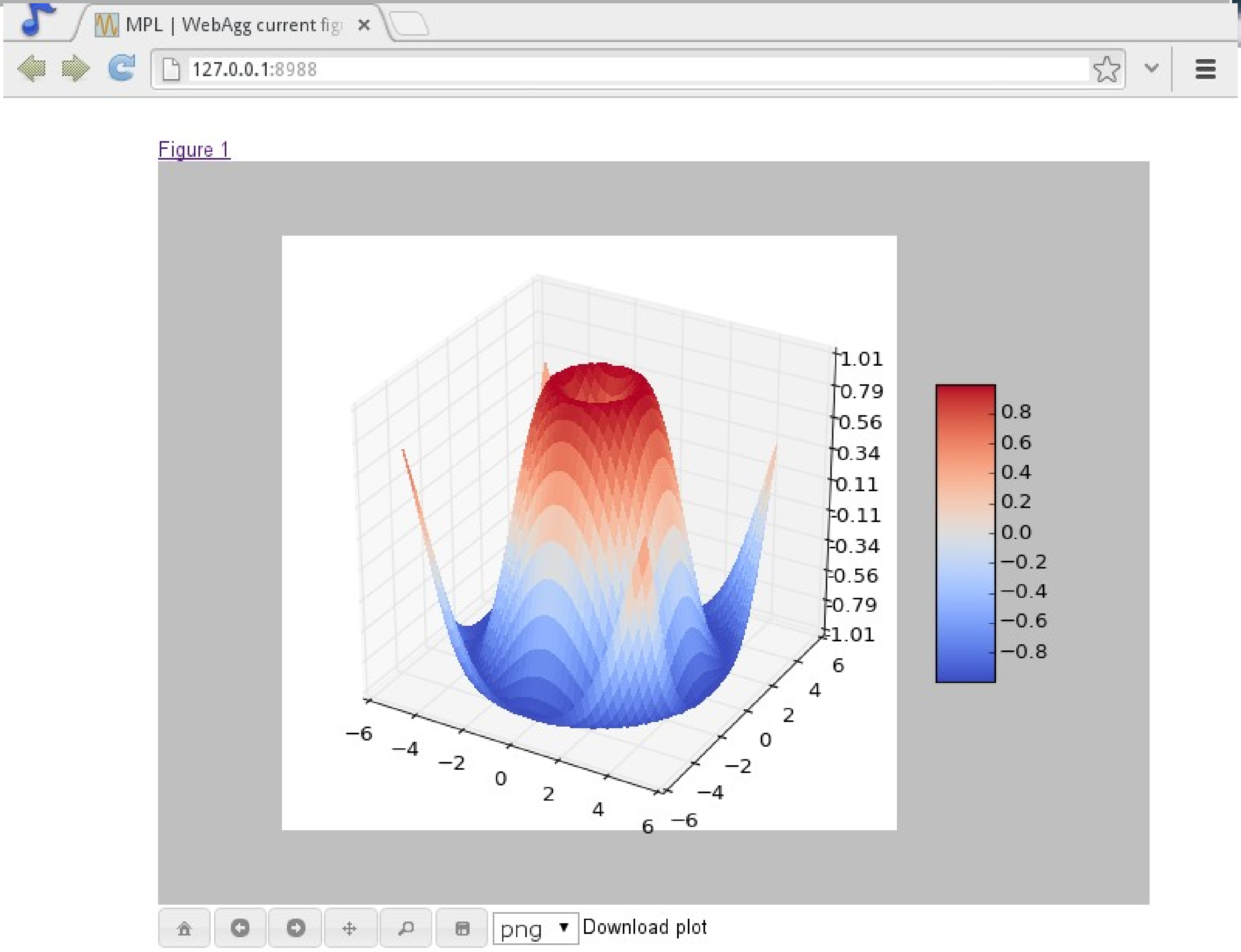
Wrapper function to pass a modified numpy grid, write out as netcdf in pydap directory, return map to embed

Display non-geographic data (eg large image)

Other options – GeoDjango?

First impression -
seems to be a lot more about vector layers than gridded data

The future – HTML5 backend for matplotlib?


MPL | WebAgg current fig
127.0.0.1:8988
Figure 1
1.01
0.79
0.56
0.34
0.11
−0.11
−0.34
−0.56
−0.79
−1.01
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−6
−4
−2
0
2
4
6
png
Download plot