cosima Documentation

Release 0.1

cosima

Jun 07, 2017

Contents:

This provides a summary of the ocean and sea ice modelling done by the COSIMA community.

It provides both examples of diagnostics to adapt to your own use and summaries of diagnostics for models run by members of the community.

The code for this cookbook is on Github: COSIMA-Cookbook

CHAPTER 1

Getting Started

COSIMA: Consortium for Ocean-Sea Ice Modelling in Australia

This repository is a gallery of self contained analysis examples.

Setting up a development environment

clone the cosima-cookbook repository:

```
git clone
```

install the cosima_cookbook utilities. From the cosima-cookbook directory, run:

```
pip install -e .
```

Run Jupyter notebook:

```
jupyter notebook --no-browser --ip='*'
```

Use:

```
scripts/jupyter_vdi.py
```

[NCI Virtual Desktop Infrastructure (VDI)](http://nci.org.au/services/vdi/)

These notebooks are designed to run on tenjin.nci.org.au which includes the VDI or on raijin.nci.org.au.

This cookbook assumes a Python computational environment and access to the NCI infrastructure.

CHAPTER 2

Diagnostics

Wind Stress

Mean zonal wind stress profile

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/xarray/core/
↪formatting.py:16: FutureWarning: The pandas.tslib module is deprecated and will be␣
↪removed in a future version.
  from pandas.tslib import OutOfBoundsDatetime
```

```
from cosima_cookbook import get_nc_variable, expts
from cosima_cookbook import memory
import matplotlib.pyplot as plt
import seaborn as sns
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/xarray/core/
↪formatting.py:16: FutureWarning: The pandas.tslib module is deprecated and will be␣
↪removed in a future version.
  from pandas.tslib import OutOfBoundsDatetime
```

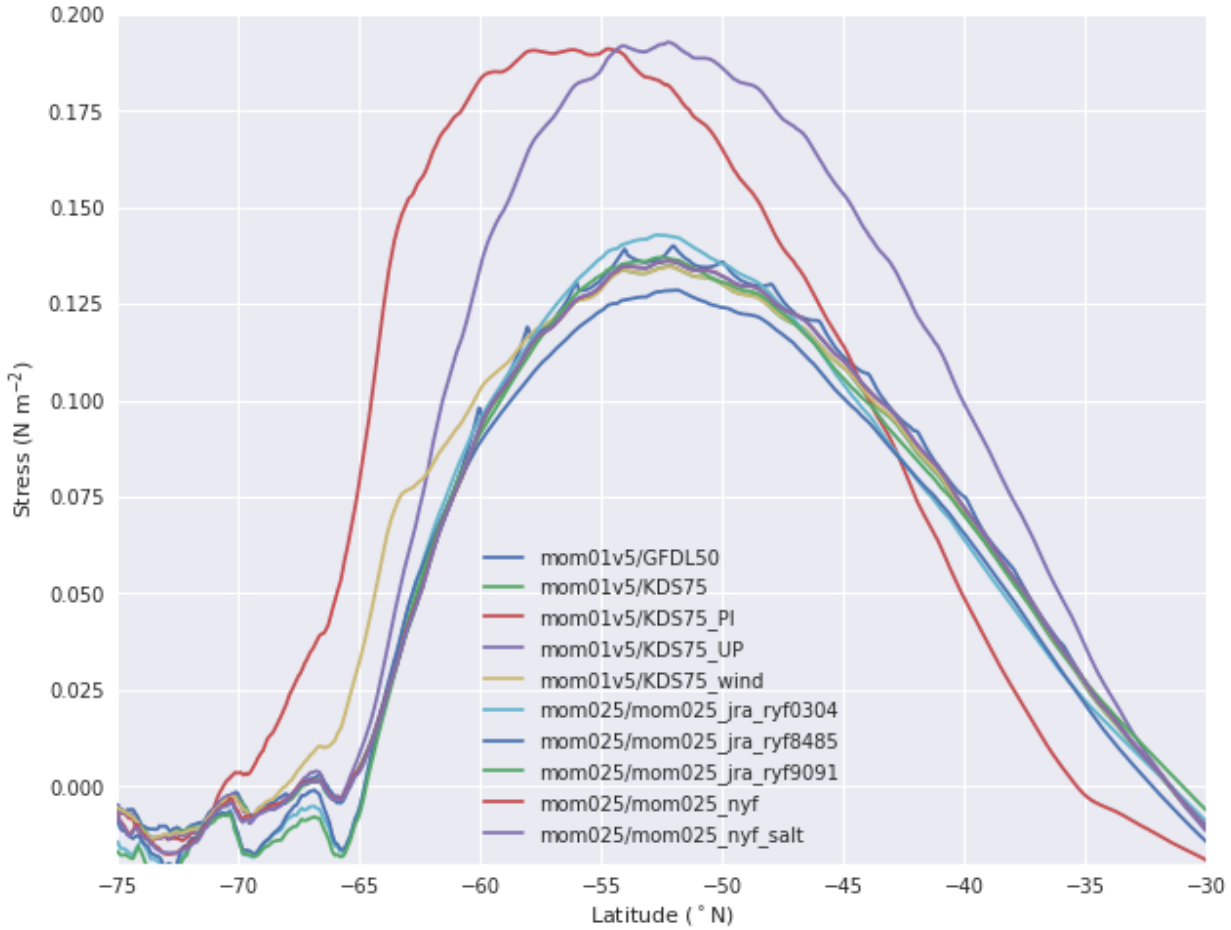
The following code block shows the zonal- and time-averaged wind stress forcing for each experiment.

```
@memory.cache
def calc_mean_tau_x(expt):

    tau_x = get_nc_variable(expt, 'ocean_month.nc', 'tau_x', n=25,
                            chunks={'xu_ocean':None})

    mean_tau_x = tau_x.mean('xu_ocean').mean('time')
```

```
    mean_tau_x = mean_tau_x.compute()
    mean_tau_x.name = expt

    return mean_tau_x
```

```
plt.figure(figsize=(10,8))

for expt in expts:
    mean_tau_x = calc_mean_tau_x(expt)
    mean_tau_x.plot()

plt.xlim([-75,-30])
plt.ylim([-0.02,0.2])
plt.xlabel('Latitude ($^\circ$N)')
plt.ylabel('Stress (N m$^{-2}$)')
plt.legend(loc=8, fontsize=10)
plt.show()
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/matplotlib/font_
↪manager.py:1297: UserWarning: findfont: Font family ['sans-serif'] not found.␣
↪Falling back to DejaVu Sans
  (prop.get_family(), self.defaultFamily[fontext]))
```



Kinetic Energy

Mean and Eddy Kinetic Energy

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
from cosima_cookbook import get_nc_variable, expts
from cosima_cookbook import memory

import matplotlib.pyplot as plt
import numpy as np
import xarray as xr
from mpl_toolkits.basemap import Basemap, shiftgrid
from tqdm import tqdm_notebook
```

We can only do this for portions of simulations which have 5-day average velocities saved, which means directories with `ocean__*.nc` files.

```
@memory.cache
def calc_eke(expt, box_index):
    yi, xi = box_index

    box = {'yu_ocean': slice(300*yi, 300*(yi+1)),
           'xu_ocean': slice(400*xi, 400*(xi+1))}

    op = lambda p: p.isel(**box)

    u = get_nc_variable(expt, 'ocean__', 'u', op=op, n=72)
    v = get_nc_variable(expt, 'ocean__', 'v', op=op, n=72)

    u_avg = u.mean('time')
    v_avg = v.mean('time')

    MKE = 0.5 * (u_avg**2 + v_avg**2)
    MKE = MKE.sum(dim='st_ocean')
    MKE = MKE.to_dataset(name='MKE')

    u_ = u - u_avg
    v_ = v - v_avg

    EKE = 0.5 * (u_**2 + v_**2)

    EKE = EKE.sum(dim='st_ocean')
    EKE = EKE.to_dataset(name='EKE')

    dsx = xr.merge([MKE, EKE])
    dsx.load()

    return dsx
```

```
from itertools import product

# ceil(x/y) = (x+y+1)//y
yi = range(2700//(300))
xi = range(3600//(400))
```

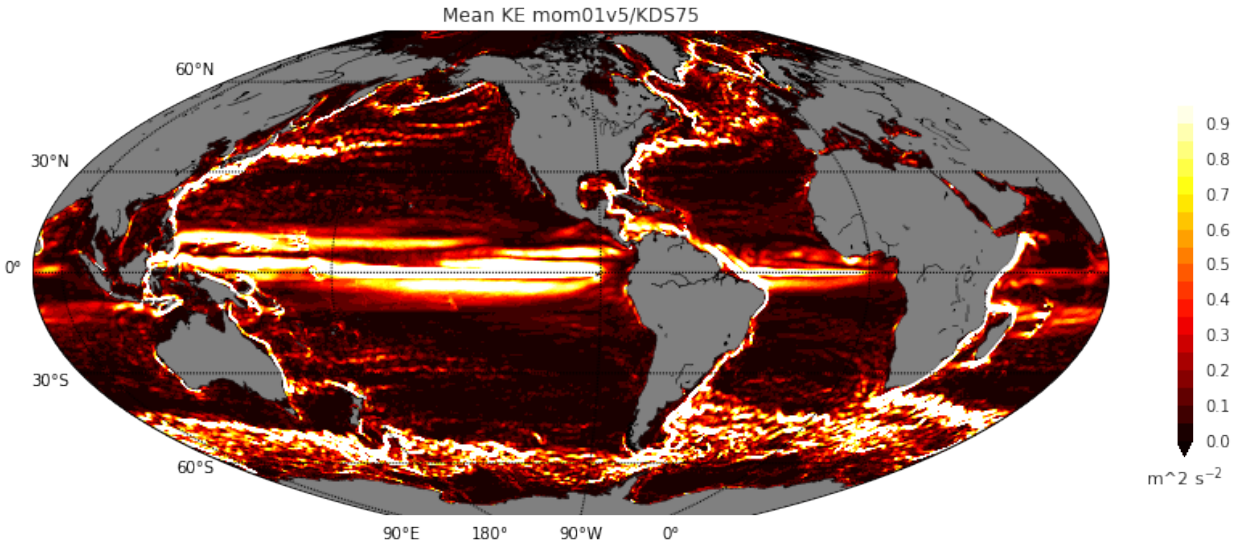
```
box_indexes = list(product(*[yi, xi]))
```

```
expt = expts[1]

print(expt)
# Plot in basemap

plt.figure(figsize=(15,6))
lev = np.arange(0, 1.0, 0.05)
map = Basemap(projection='mbtfpq',
              lon_0 = -100, resolution='l')
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
map.drawparallels(np.arange(-60.,61.,30.),
                  labels=[True,False,False,False])
map.drawmeridians(np.arange(-180.,181.,90.),
                  labels=[False,False,False,True])

for box_index in tqdm_notebook(box_indexes):
    #print(box_index)
    dsx = calc_eke(expt, box_index)

    x=dsx.xu_ocean[:]
    y=dsx.yu_ocean[:]
    lon, lat = np.meshgrid(x, y)

    X, Y = map(lon,lat)

    map.contourf(X, Y, dsx.MKE,
                 cmap=plt.cm.hot,
                 levels=lev,extend='both')

cb = plt.colorbar(orientation='vertical',shrink = 0.7)

cb.ax.set_xlabel('m^2 s$^{-2}$')
plt.title('Mean KE {}'.format(expt))
```

```
mom01v5/KDS75
```

```
<matplotlib.text.Text at 0x7f21a2e5b668>
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/matplotlib/font_
↪manager.py:1297: UserWarning: findfont: Font family ['sans-serif'] not found.␣
↪Falling back to DejaVu Sans
  (prop.get_family(), self.defaultFamily[fontext]))
```



Zonal Mean Temperature

Calculating the zonal mean of a quantity

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
from cosima_cookbook import get_nc_variable, expts
from cosima_cookbook import memory

import matplotlib.pyplot as plt
```

```
@memory.cache
def calc_zonal_mean_temp(expt):
    print('Calculating {} zonal_mean_temp'.format(expt))

    if expt == 'mom01v5/KDS75':
        ncfile = 'ocean_month.nc'
    else:
        ncfile = 'ocean.nc'

    zonal_temp = get_nc_variable(expt, ncfile, 'temp',
                                 chunks={'st_ocean': None},
                                 n=25)

    zonal_mean_temp = zonal_temp.mean('xt_ocean').mean('time')
    zonal_mean_temp.load()

    return zonal_mean_temp
```

```
def plot_zonal_mean_temp(expt):
    zonal_mean_temp = calc_zonal_mean_temp(expt)

    zonal_mean_temp.plot()
```

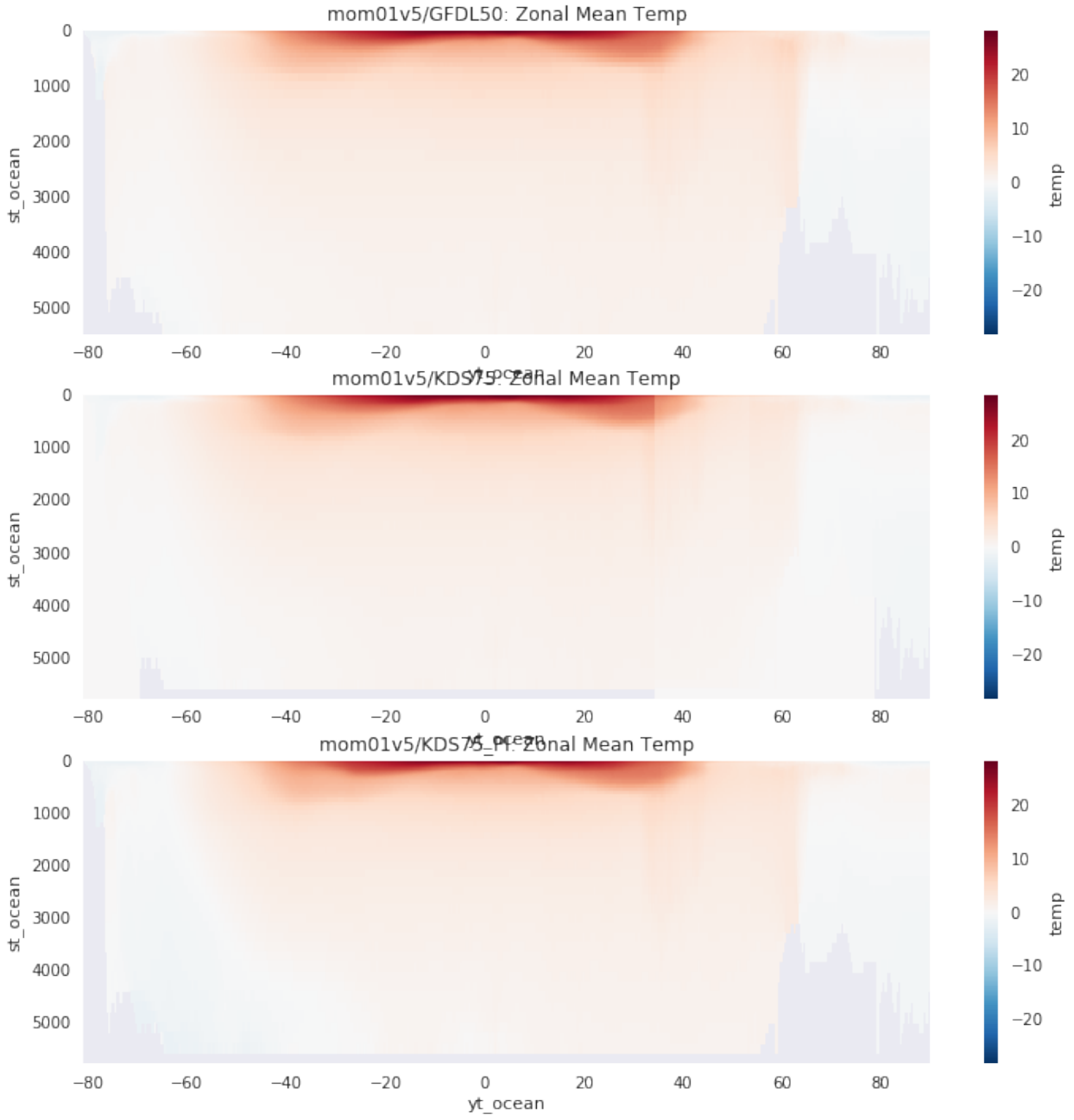
```
    plt.gca().invert_yaxis()
    plt.title('{}: Zonal Mean Temp'.format(expt))
```

```
plt.figure(figsize=(12,12))
gs = gridspec.GridSpec(3, 1)
for i, expt in enumerate(expts):

    plt.subplot(gs[i])

    plot_zonal_mean_temp(expt)
```

```
Calculating mom01v5/GFDL50 zonal_mean_temp
Calculating mom01v5/KDS75 zonal_mean_temp
Calculating mom01v5/KDS75_PI zonal_mean_temp
```

```
---------------------------------------------------------------------------

IndexError                                Traceback (most recent call last)

<ipython-input-23-86fa6fb63660> in <module>()
      3 for i, expt in enumerate(expts):
      4
----> 5     plt.subplot(gs[i])
      6
      7     plot_zonal_mean_temp(expt)


/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/matplotlib/gridspec.
↪py in __getitem__(self, key)
    182                     key += total
    183                 if key >= total or key < 0 :
--> 184                     raise IndexError("index out of range")
    185                 num1, num2 = key, None
    186


IndexError: index out of range
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/matplotlib/font_
↪manager.py:1297: UserWarning: findfont: Font family ['sans-serif'] not found.␣
↪Falling back to DejaVu Sans
  (prop.get_family(), self.defaultFamily[fontext]))
```



Major Transports

Transport diagnostics from each of the major straits.

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
from cosima_cookbook import get_nc_variable, expts, build_index
from cosima_cookbook import memory
```

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

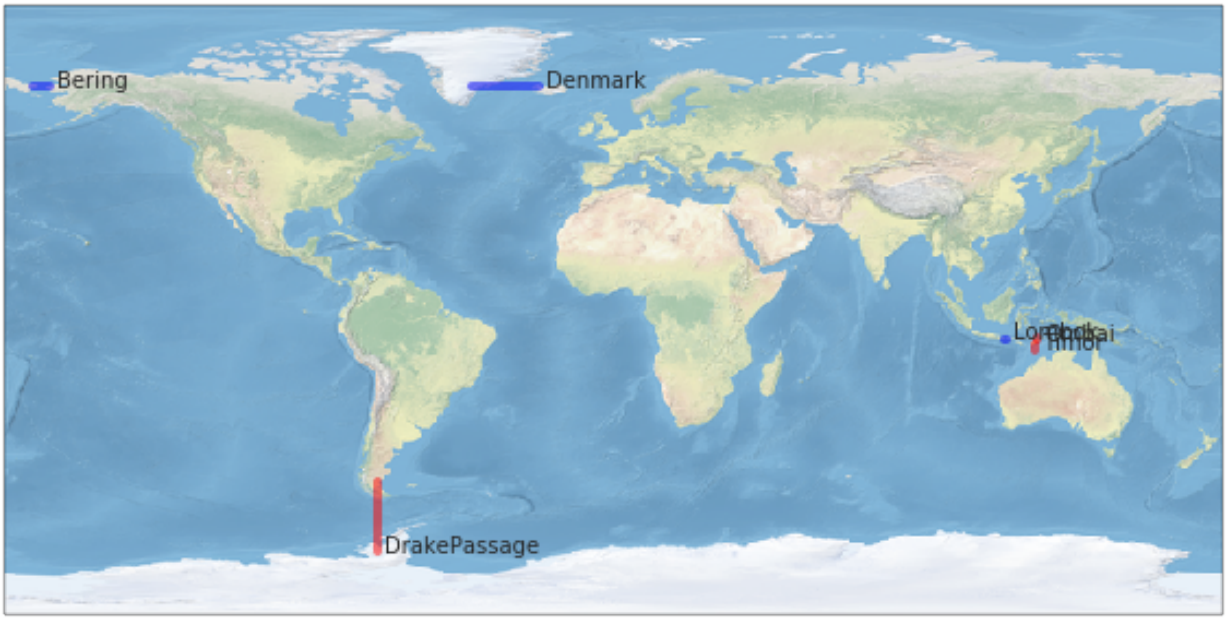
```
straights = [ {'name': 'DrakePassage', 'xloc':2100,'ymin':225,'ymax':650},
              {'name': 'Lombok',       'yloc':1158,'xmin':354,'xmax':361},
              {'name': 'Ombai',        'xloc':449,'ymin':1152,'ymax':1163},
              {'name': 'Timor',        'xloc':440,'ymin':1125,'ymax':1145},
              {'name': 'Bering',       'yloc':2125,'xmin':1080,'xmax':1130},
              {'name': 'Denmark',      'yloc':2125,'xmin':2380,'xmax':2580},
            ]
```

```
xu_ocean = get_nc_variable('mom01v5/KDS75', 'ocean_grid', 'xu_ocean', n=1).
↪isel(time=0)
yu_ocean = get_nc_variable('mom01v5/KDS75', 'ocean_grid', 'yu_ocean', n=1).
↪isel(time=0)
```

```
import cartopy.crs as ccrs

plt.figure(figsize=(10,8))
ax = plt.axes(projection=ccrs.PlateCarree())

for straight in straights:

    if 'xloc' in straight:
        xloc = straight['xloc']
        ymin = straight['ymin']
        ymax = straight['ymax']

        plt.plot([xu_ocean[xloc], xu_ocean[xloc]],
                 [yu_ocean[ymin], yu_ocean[ymax]],

            color='red', linewidth=4,alpha=0.5,
             transform=ccrs.Geodetic(),
         )

        plt.text(xu_ocean[xloc]+2, yu_ocean[ymin], straight['name'],
         horizontalalignment='left',
        transform=ccrs.Geodetic()
        )
    else:
        yloc = straight['yloc']
        xmin = straight['xmin']
        xmax = straight['xmax']

        plt.plot([xu_ocean[xmin], xu_ocean[xmax]],
                 [yu_ocean[yloc], yu_ocean[yloc]],

             color='blue', linewidth=4,alpha=0.5,
             transform=ccrs.Geodetic(),
         )

        plt.text(xu_ocean[xmax]+2, yu_ocean[yloc], straight['name'],
         horizontalalignment='left',
        transform=ccrs.Geodetic()
        )
```

```
ax.stock_img()
ax.set_global()
plt.show()
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/matplotlib/font_
↪manager.py:1297: UserWarning: findfont: Font family ['sans-serif'] not found.␣
↪Falling back to DejaVu Sans
  (prop.get_family(), self.defaultFamily[fontext]))
```




```
@memory.cache
def calc_transport(expt, straight):
    ## Function to calculate barotropic transport across a given line of latitude or␣
↪longitude.
    ## Primarily designed for flow through straits.

    print('Calculating {}:{} transport'.format(expt, straight['name']))

    if 'xloc' in straight:

        ymin, ymax = straight['ymin'], straight['ymax']
        xloc = straight['xloc']

        op = lambda p: p.isel(xu_ocean=xloc)\
                        .isel(yt_ocean=slice(ymin, ymax))\
                        .chunk({'time':1})\
                        .sum('st_ocean').sum('yt_ocean')

        transport = get_nc_variable(expt, 'ocean.nc', 'tx_trans',
                                    chunks=None,
                                    op=op)

    elif 'yloc' in straight:

        xmin, xmax = straight['xmin'], straight['xmax']
        yloc = straight['yloc']
```

```
        op = lambda p: p.isel(yu_ocean=yloc)\
                        .isel(xt_ocean=slice(xmin, xmax))\
                        .chunk({'time':1})\
                        .sum('st_ocean').sum('xt_ocean')

        transport = get_nc_variable(expt, 'ocean.nc', 'ty_trans',
                                    chunks=None,
                                    op=op)
    else:
        print('Transports must be along lines of either constant latitude or longitude
↪')
        return None

    transport = transport.load()

    return transport
```

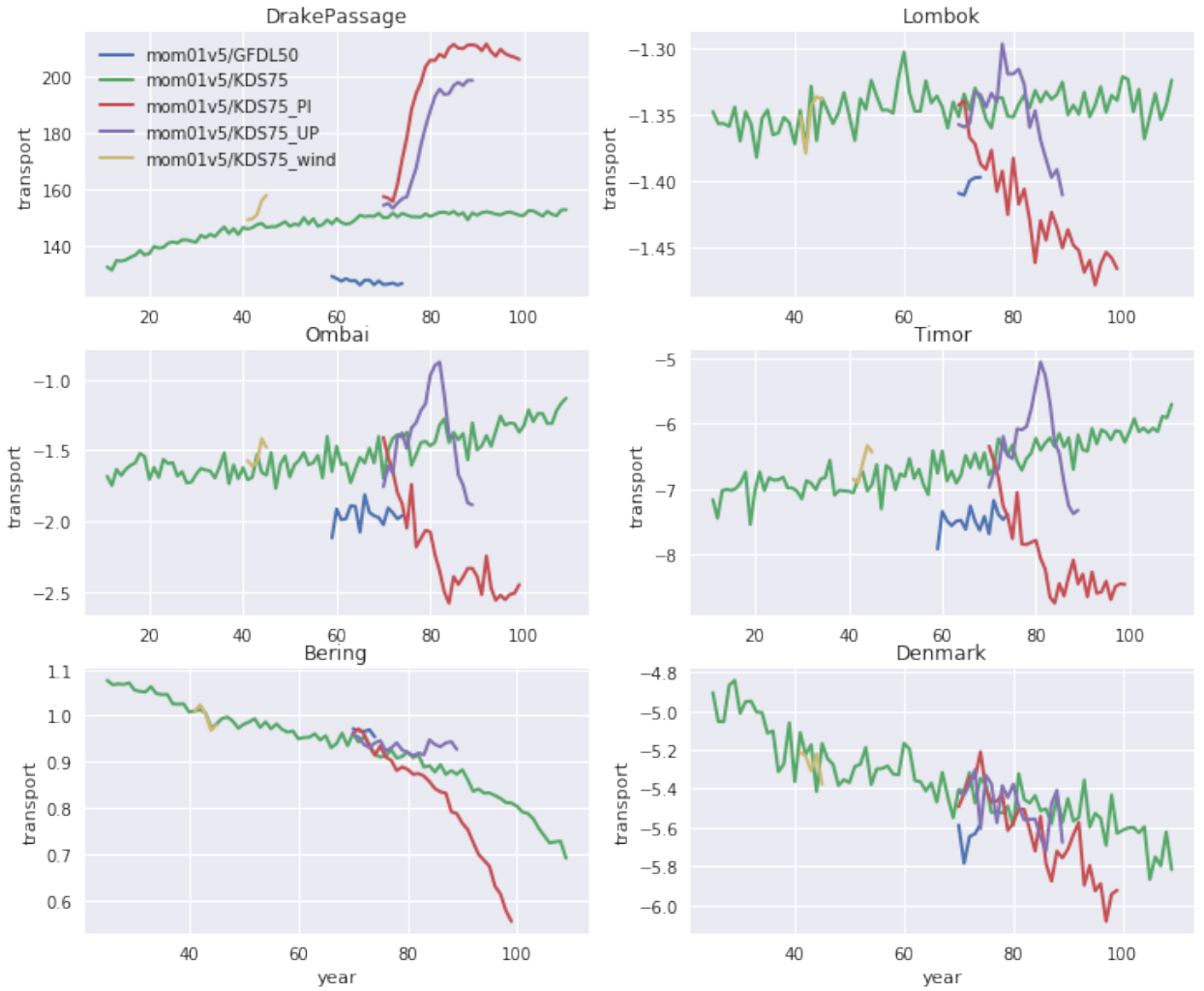
When distributing a computing with dask, you should try and minimize the number of task while keeping each task appropriate to a single work regarding memory usage.

```
plt.figure(figsize=(12,10))

for expt in expts:
    if 'mom025' in expt:
        # need to change straights from grid coordinates to real word positions
        continue

    nplot = 0

    for straight in straights:
        nplot += 1

        transport = calc_transport(expt, straight)

        # see https://github.com/spencerahill/aospy/issues/98#issuecomment-256043833
        da = transport.to_dataset(name='transport')

        periods = []
        for date in da['time']:
            raw_date = date.values.item()
            periods.append(pd.Period(year=raw_date.year, month=raw_date.month,
                                     day=raw_date.day, freq='D'))
        da['time'] = pd.PeriodIndex(periods)
        da = da.groupby('time.year').mean('time')
        transport = da.transport

        plt.subplot(3, 2, nplot)
        transport.plot(label=expt, linewidth=2)
        plt.title( straight['name'])

plt.subplot(321)
plt.legend(loc='upper left')
#plt.ylabel('Transport (Sv)')
plt.xlabel('')
plt.subplot(322)
#plt.ylabel('')
plt.xlabel('')
```

```
plt.subplot(323)
#plt.ylabel('Transport (Sv)')
plt.xlabel('')
plt.subplot(324)
#plt.ylabel('')
plt.xlabel('')
plt.subplot(325)
#plt.ylabel('Transport (Sv)')
#plt.xlabel('Time')
plt.subplot(326)
#plt.ylabel('')
#plt.xlabel('Time')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f636fedb5f8>
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/matplotlib/font_
↪manager.py:1297: UserWarning: findfont: Font family ['sans-serif'] not found.␣
↪Falling back to DejaVu Sans
  (prop.get_family(), self.defaultFamily[fontext]))
```



Scalar Quantities

Globally averaged quantities

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
from cosima_cookbook import build_index, expts
from cosima_cookbook import memory

import netCDF4
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import dask.bag as db
```

```
def extract_data(f):
    dataset = netCDF4.Dataset(f)

    data = {}
    for v in dataset.variables.values():
        if 'time' in v.dimensions and 'scalar_axis' in v.dimensions:
            data[v.name] = v[:].flatten()
    index = pd.to_datetime(dataset['time'][:], unit='D')
    df = pd.DataFrame(index=index, data=data)
    return df

@memory.cache
def calc_scalar_quantities(expt):
    df = build_index(expt)

    # identify all scalar variables
    df = df[(df.ncfile == 'ocean_scalar.nc') & (df.dimensions == ('time', 'scalar_axis
↪' )) ]
    ncfiles = sorted(list(df.path))

    b = db.from_sequence(ncfiles)
    b = list(b.map(extract_data))
    df = pd.concat(b)

    return df
```

```
scalar_quantities = {}
for expt in expts:
    print(expt)
    scalar_quantities[expt] = calc_scalar_quantities(expt)
```

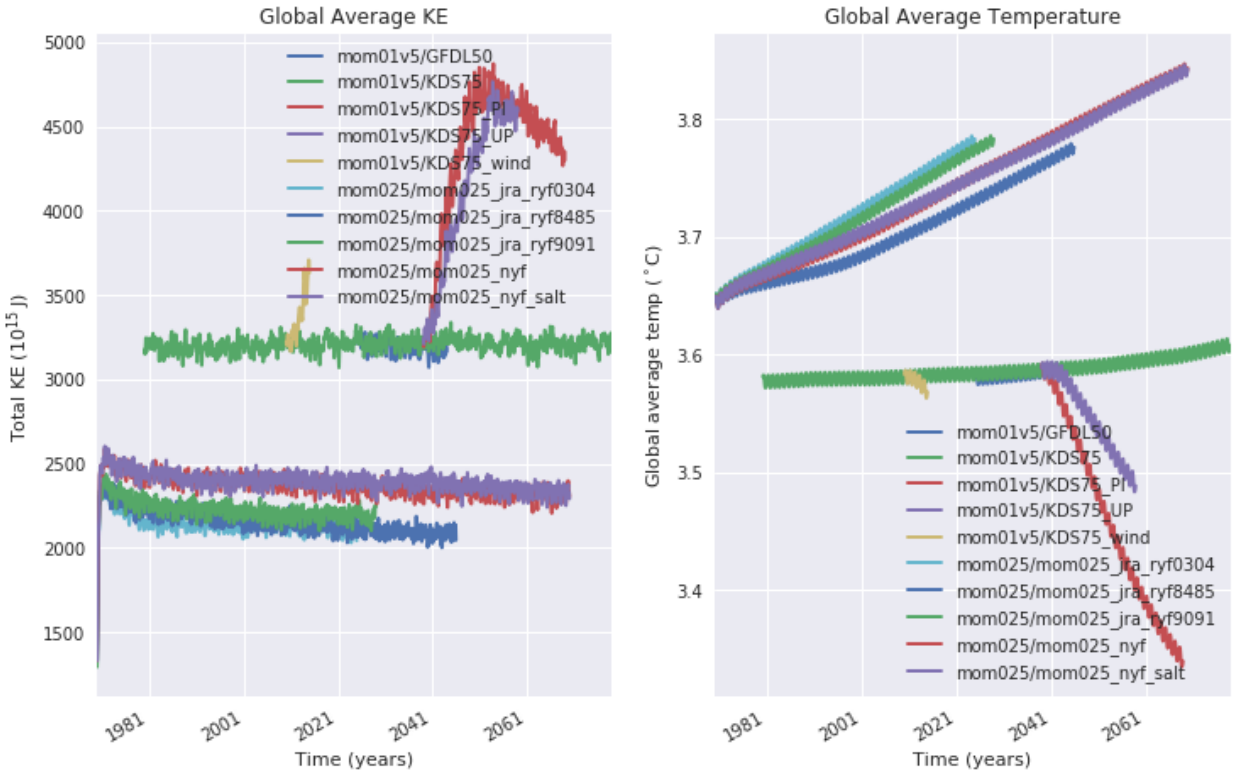
```
scalar_quantities['mom01v5/GFDL50'].describe().T
```

```
plt.figure(figsize=(12,8))

for expt in expts:

    plt.subplot(121)
    scalar_quantities[expt].ke_tot.plot(label=expt, linewidth=2)
```

```
    plt.subplot(122)
    scalar_quantities[expt].temp_global_ave.plot(label=expt, linewidth=2)

plt.subplot(121)
plt.legend(loc='upper right')
plt.ylabel('Total KE (10$^{15}$ J)')
plt.xlabel('Time (years)')
plt.title('Global Average KE')

plt.subplot(122)
plt.legend(loc='lower right')
plt.ylabel('Global average temp ($^\circ$C)')
plt.xlabel('Time (years)')
plt.title('Global Average Temperature')
```

```
<matplotlib.text.Text at 0x7fb7effbd390>
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/matplotlib/font_
↪manager.py:1297: UserWarning: findfont: Font family ['sans-serif'] not found.␣
↪Falling back to DejaVu Sans
  (prop.get_family(), self.defaultFamily[fontext]))
```



Overturning Circulation

Overturning circulation in density space using ty_trans_rho.

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
/mnt/miniconda3/envs/cosima/lib/python3.6/site-packages/xarray/core/formatting.py:16:␣
↪FutureWarning: The pandas.tslib module is deprecated and will be removed in a␣
↪future version.
  from pandas.tslib import OutOfBoundsDatetime
```

```
from cosima_cookbook import get_nc_variable, expts
from cosima_cookbook import memory
import matplotlib.pyplot as plt
import numpy as np
```

```
/mnt/miniconda3/envs/cosima/lib/python3.6/site-packages/xarray/core/formatting.py:16:␣
↪FutureWarning: The pandas.tslib module is deprecated and will be removed in a␣
↪future version.
  from pandas.tslib import OutOfBoundsDatetime
```

We will zonally average this diagnostic, without accounting for the tripolar grid, so ignore the Arctic.

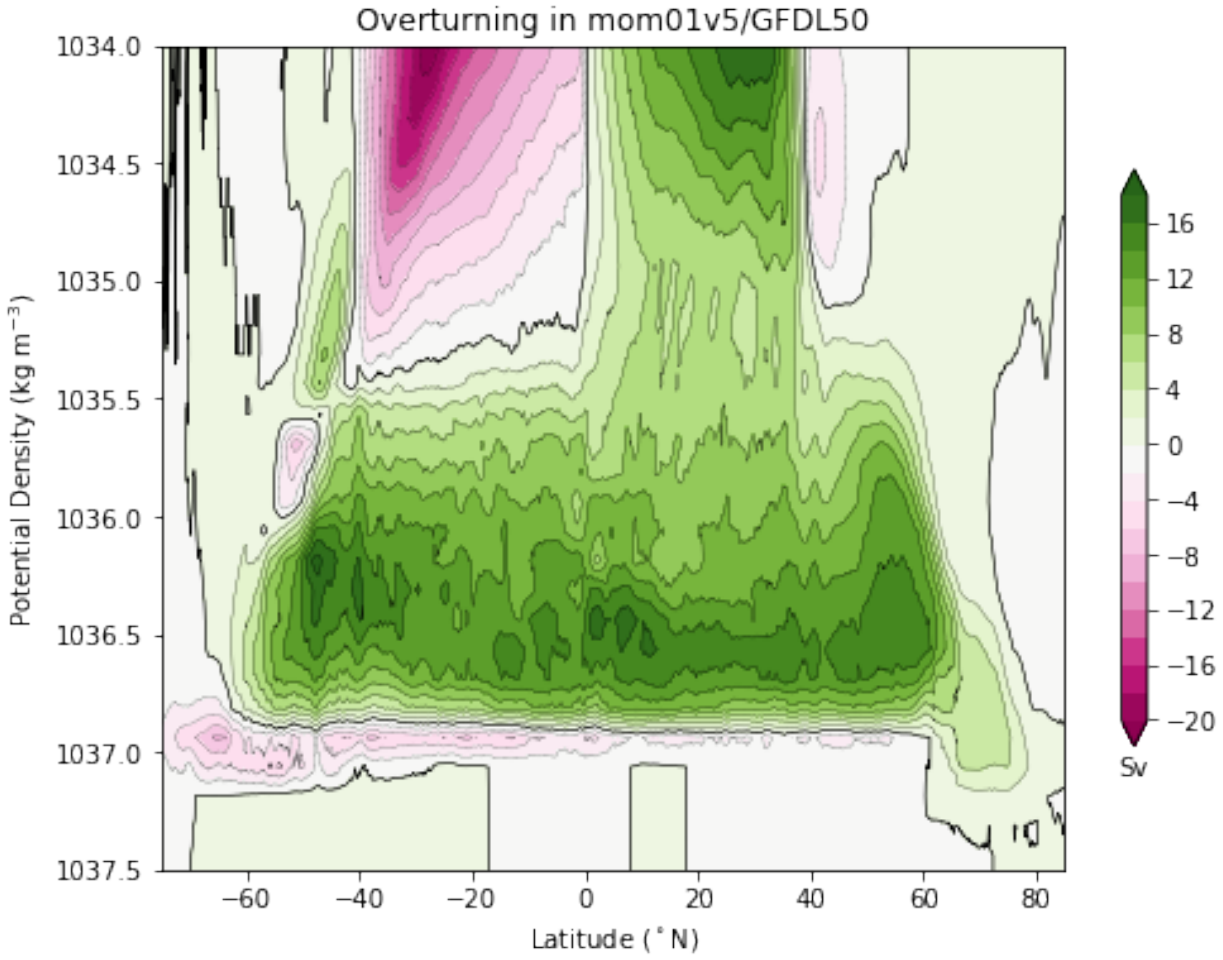
```
@memory.cache
def calc_psi_avg(expt):
    print('Calculating {} psi_avg'.format(expt))

    op = lambda p: p.sum('grid_xt_ocean').cumsum('potrho')

    psi = get_nc_variable(expt, 'ocean.nc', 'ty_trans_rho',
                          op=op,
                          chunks={'potrho': None}, n=5)

    psi_avg = psi.mean('time')
    psi_avg = psi_avg.compute()

    return psi_avg
```

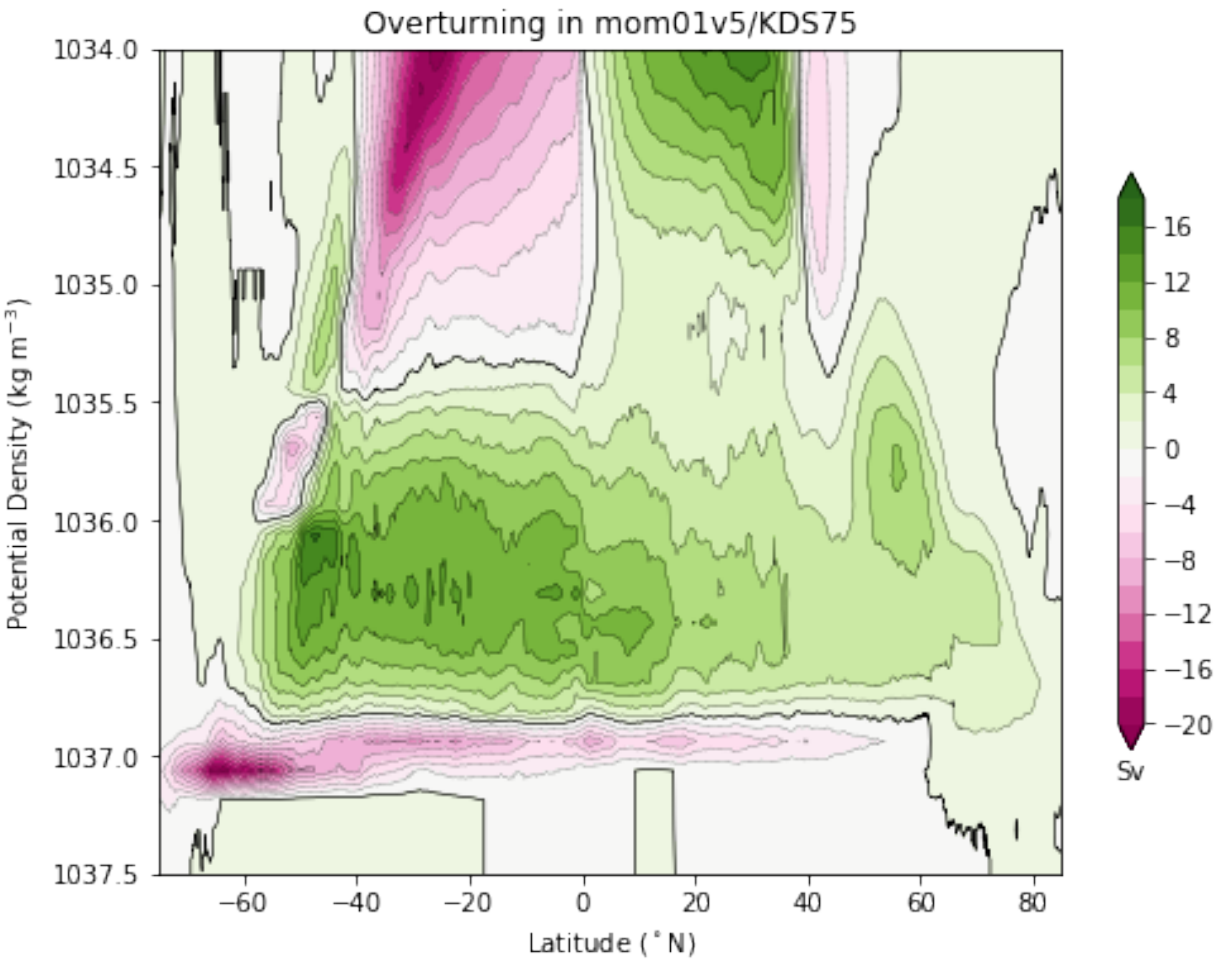
```
def plot_psi(psi_avg, expt, clev=np.arange(-20,20,2)):

    plt.contourf(psi_avg.grid_yu_ocean,
                 psi_avg.potrho,
                 psi_avg,
                 cmap=plt.cm.PiYG,levels=clev,extend='both')
    cb=plt.colorbar(orientation='vertical', shrink = 0.7)
    cb.ax.set_xlabel('Sv')
    plt.contour(psi_avg.grid_yu_ocean,
                psi_avg.potrho,
                psi_avg, levels=clev, colors='k', linewidths=0.25)
    plt.contour(psi_avg.grid_yu_ocean,
                psi_avg.potrho, psi_avg,
                levels=[0.0,], colors='k', linewidths=0.5)
    plt.gca().invert_yaxis()

    plt.ylim((1037.5,1034))
    plt.ylabel('Potential Density (kg m$^{-3}$)')
    plt.xlabel('Latitude ($^\circ$N)')
    plt.xlim([-75,85])
```

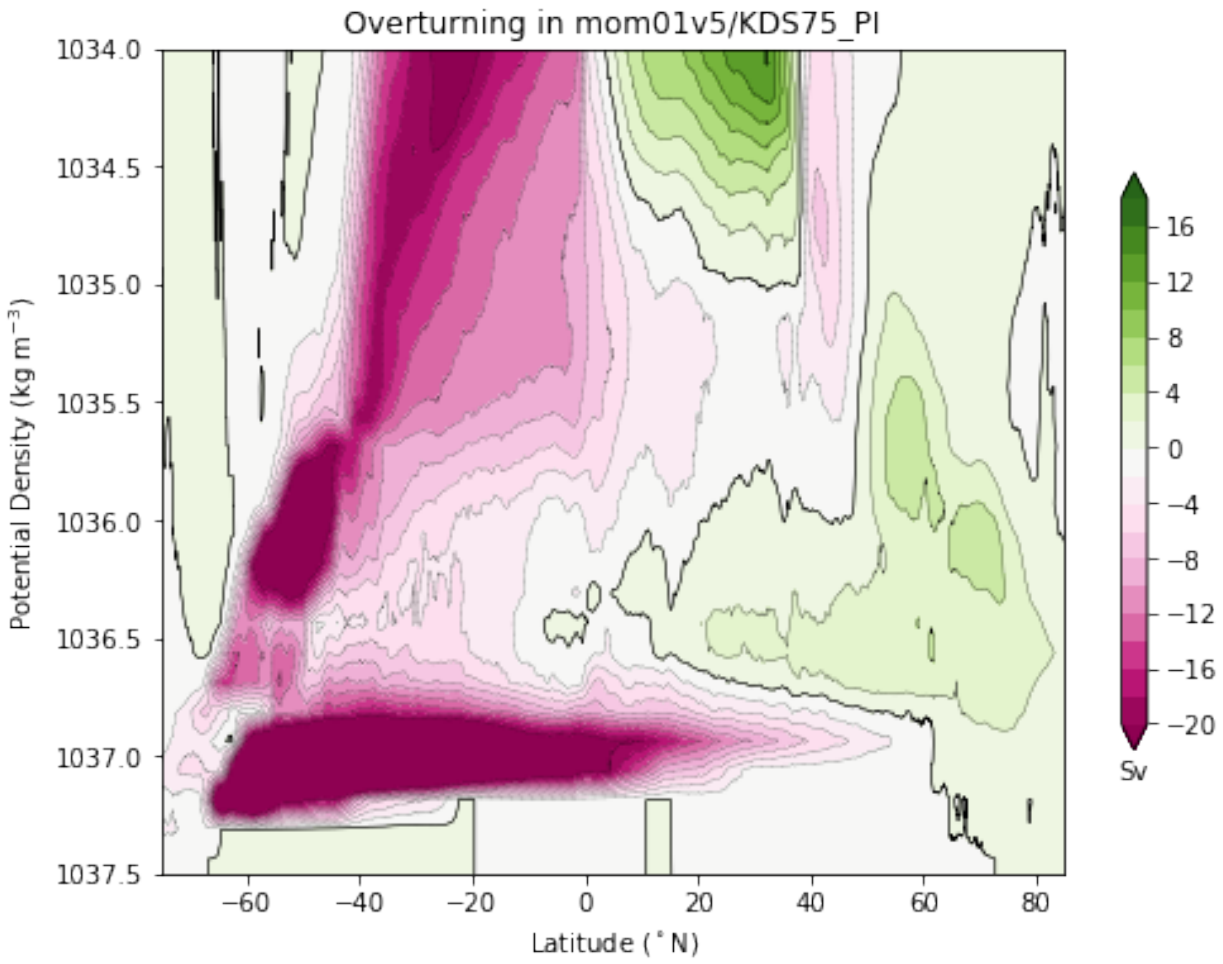
```
    plt.title('Overturning in %s' % expt)
```

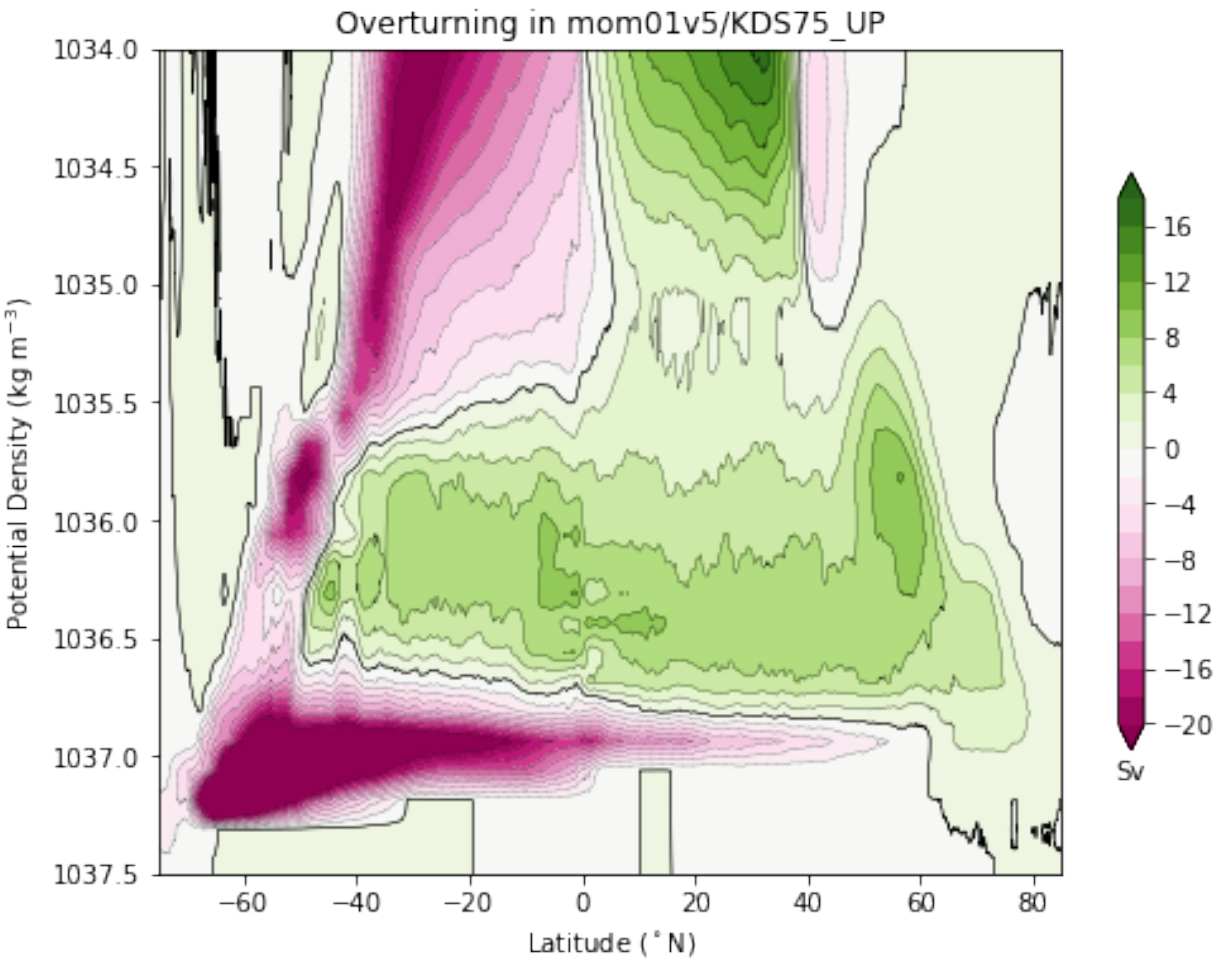
```
for expt in expts:
    plt.figure(figsize=(8, 6))
    psi_avg = calc_psi_avg(expt)
    plot_psi(psi_avg, expt)
```

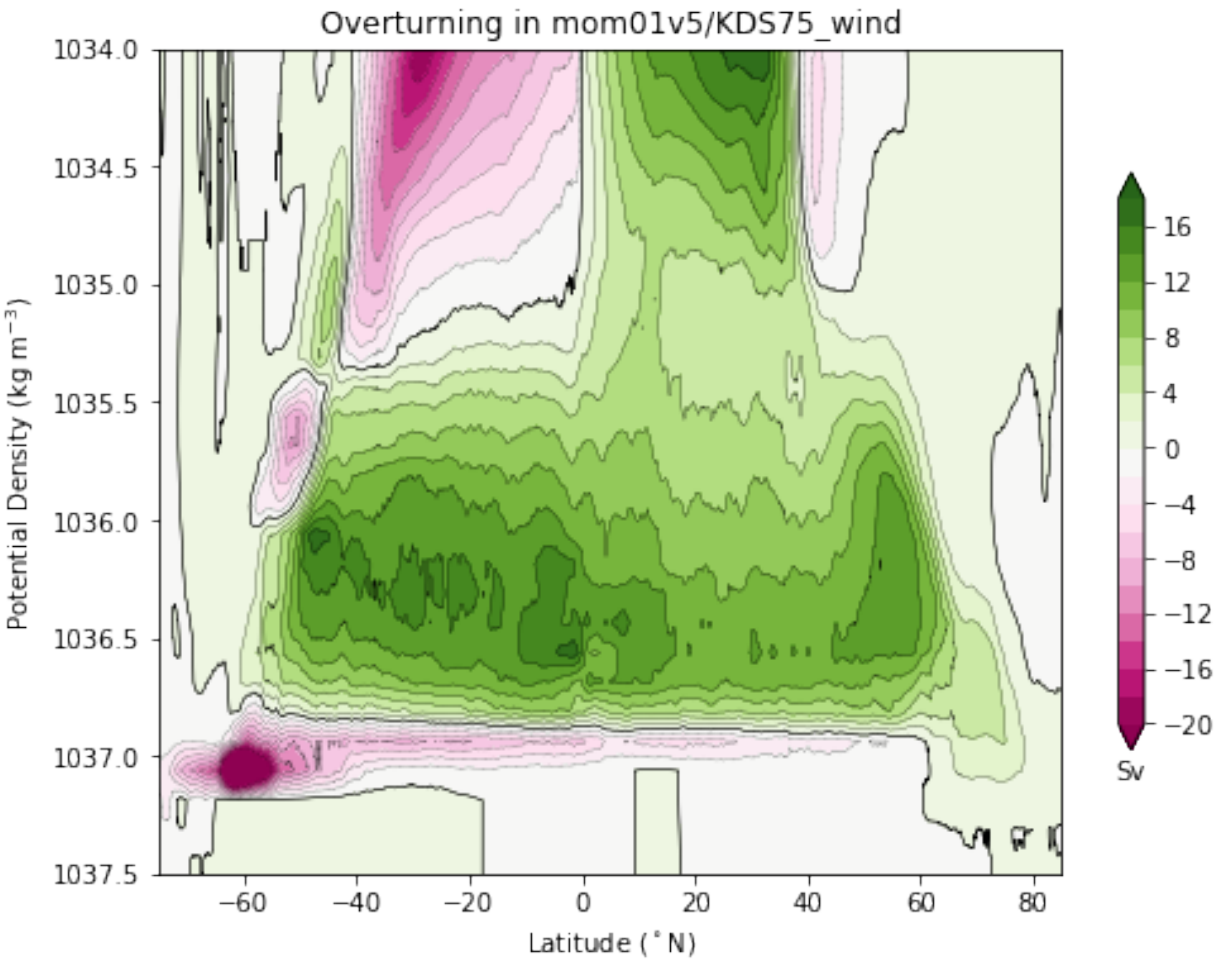
```
Calculating mom01v5/KDS75_PI psi_avg
Calculating mom01v5/KDS75_UP psi_avg
Calculating mom01v5/KDS75_wind psi_avg
Calculating mom025/mom025_jra_ryf0304 psi_avg
Calculating mom025/mom025_jra_ryf8485 psi_avg
Calculating mom025/mom025_jra_ryf9091 psi_avg
Calculating mom025/mom025_nyf psi_avg
Calculating mom025/mom025_nyf_salt psi_avg
```

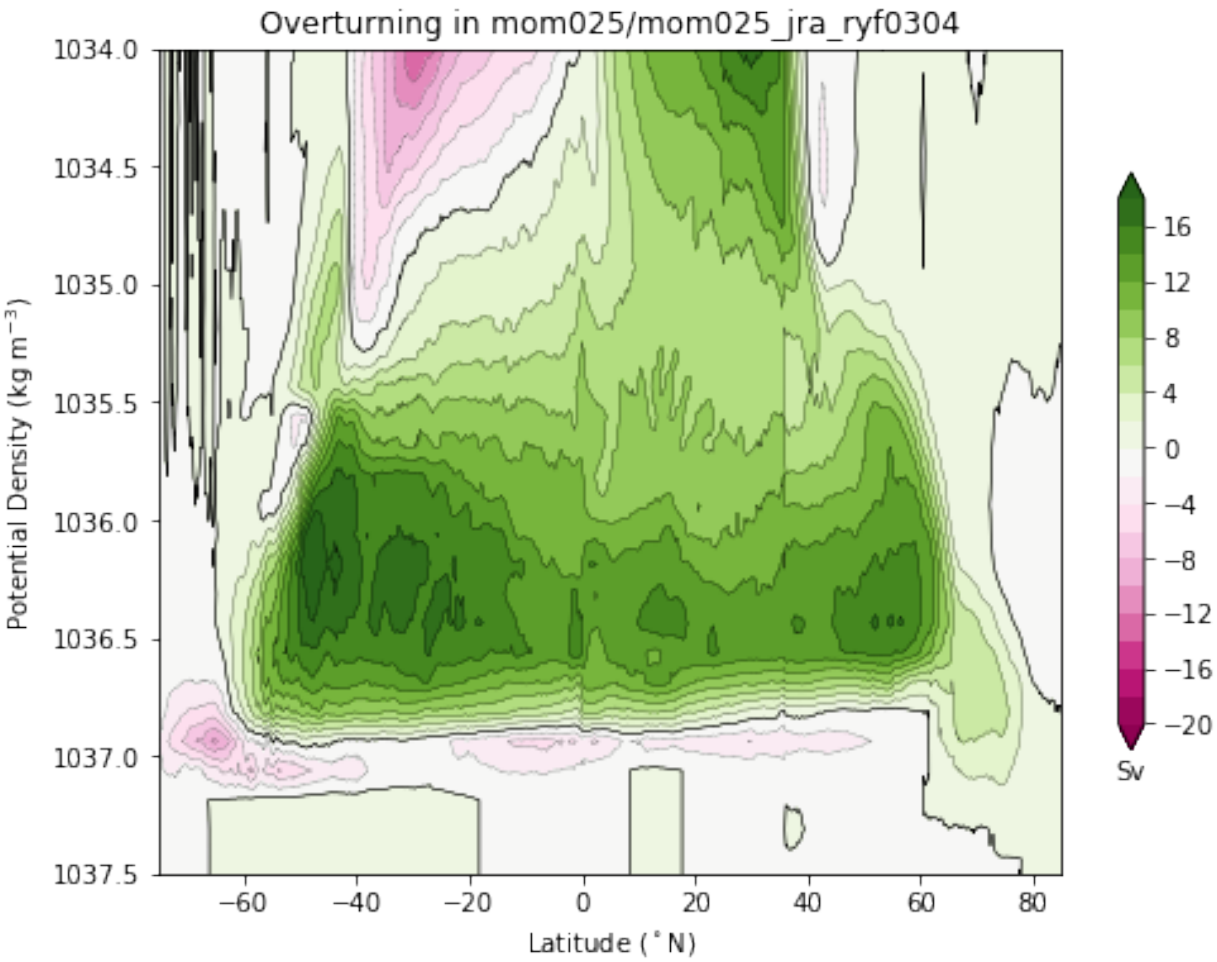


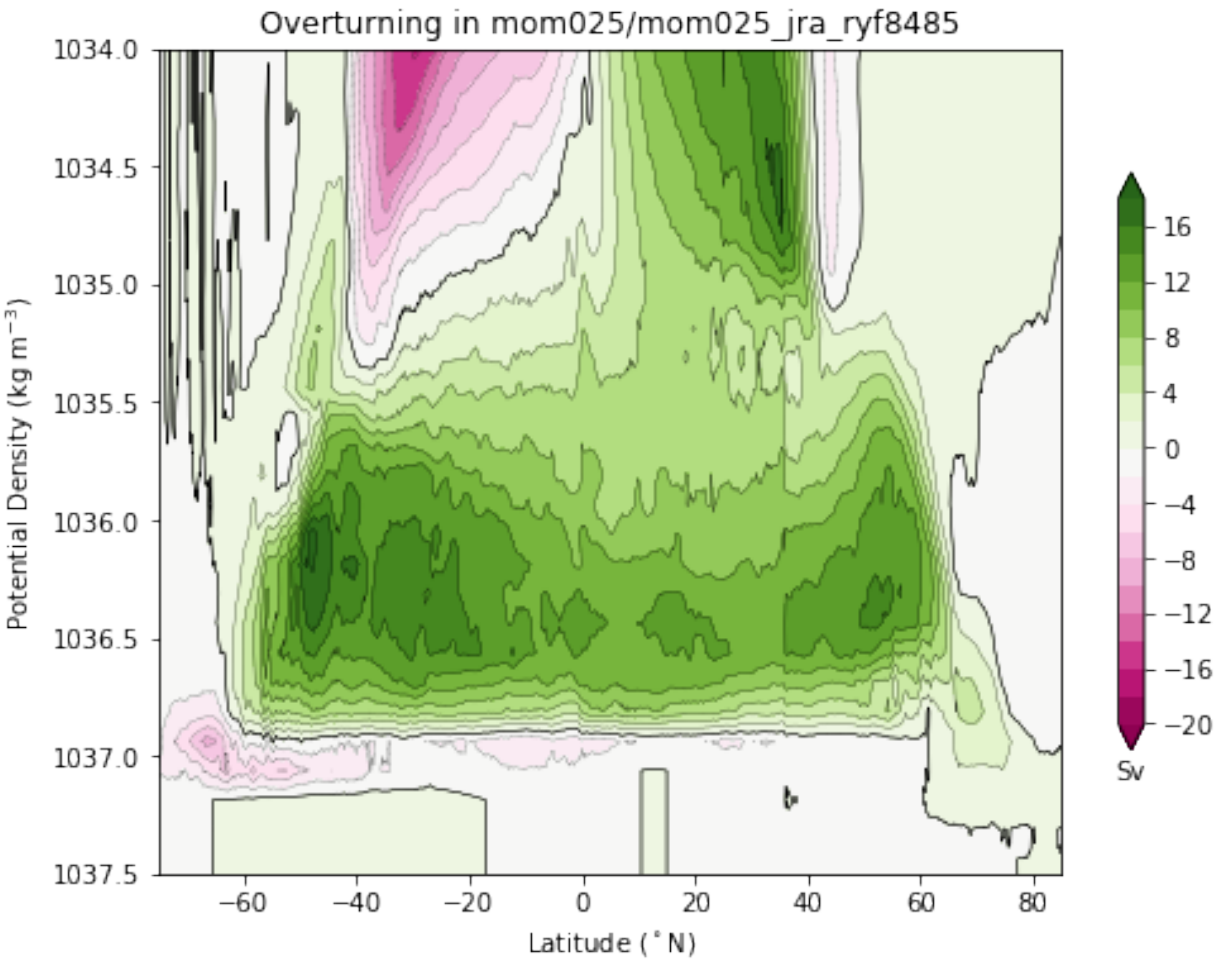

Overturning in mom01v5/KDS75
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv

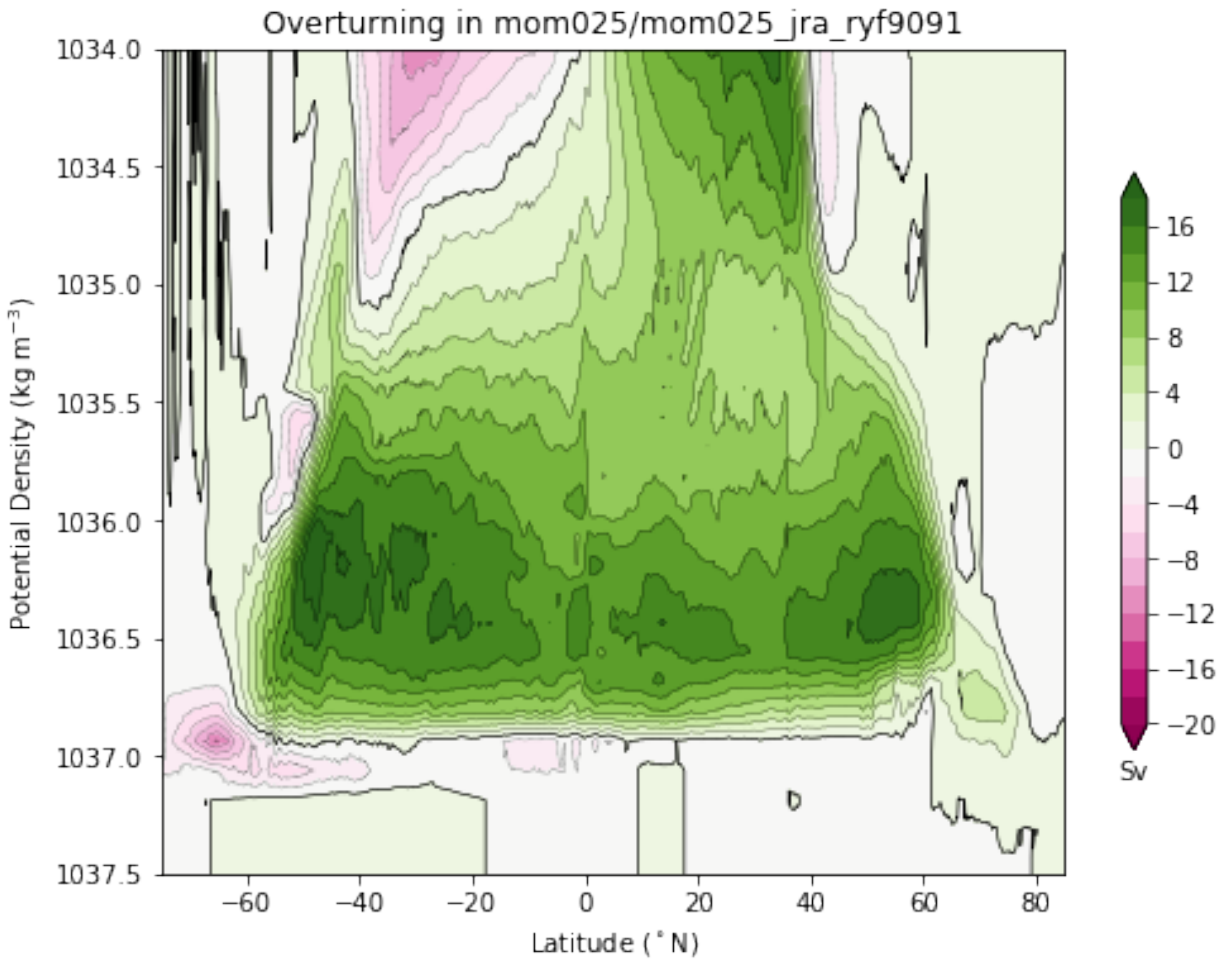

Overturning in mom01v5/KDS75_PI
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv

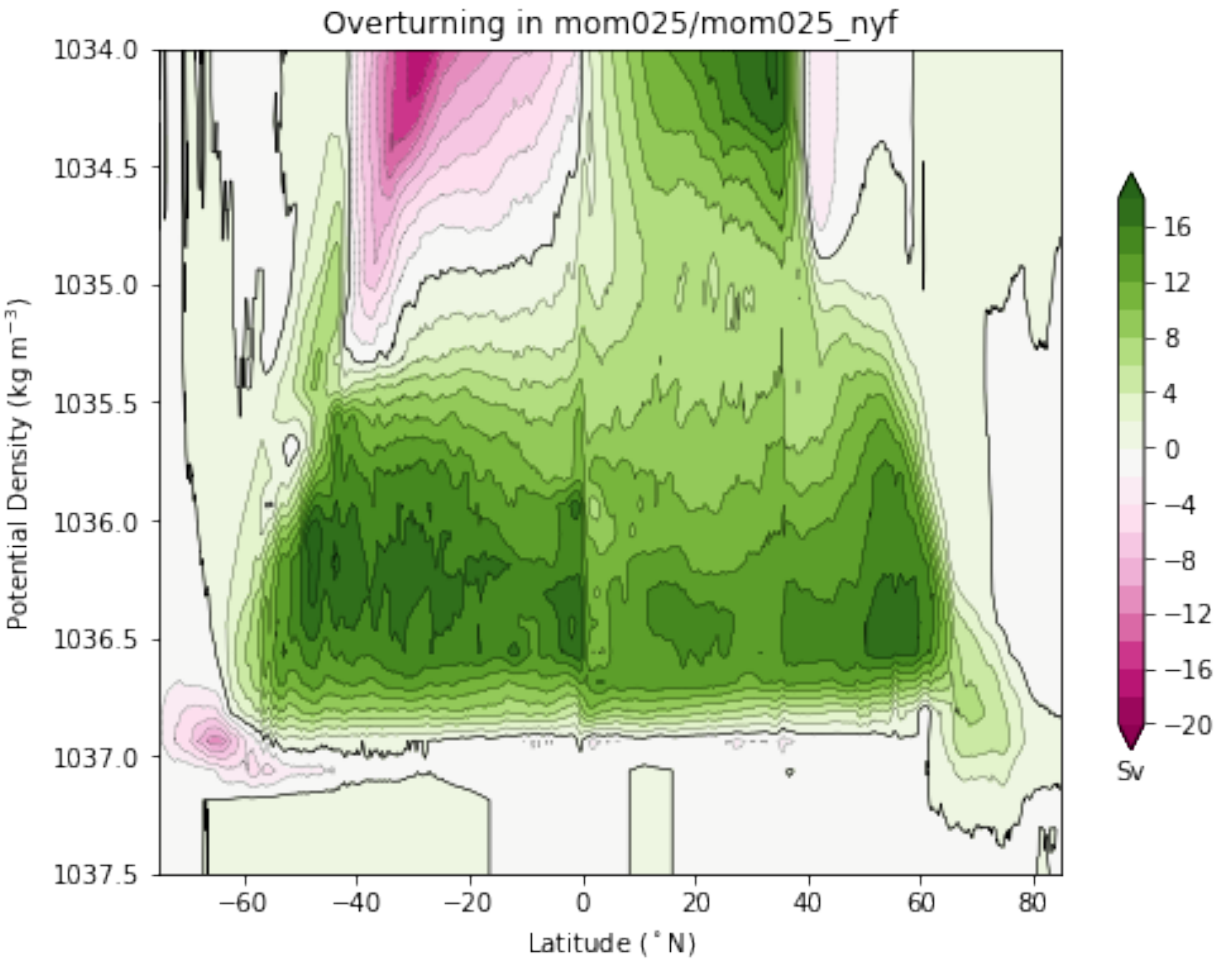

Overturning in mom01v5/KDS75_UP
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv

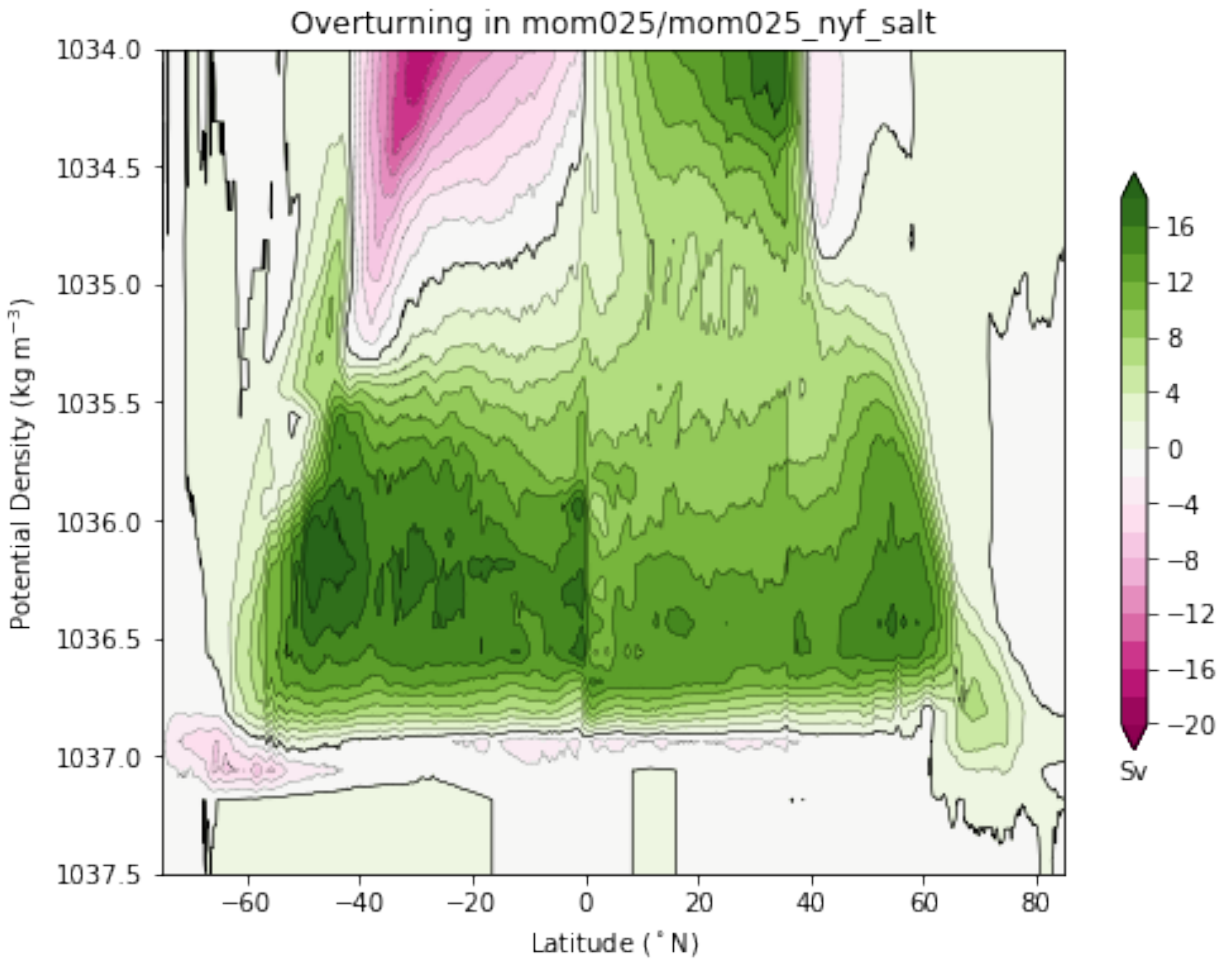

Overturning in mom01v5/KDS75_wind
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv


Overturning in mom025/mom025_jra_ryf0304
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv


Overturning in mom025/mom025_jra_ryf8485
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv


Overturning in mom025/mom025_jra_ryf9091
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv


Overturning in mom025/mom025_nyf
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv


Overturning in mom025/mom025_nyf_salt
Potential Density (kg m^{-3})
1034.0
1034.5
1035.0
1035.5
1036.0
1036.5
1037.0
1037.5
−60
−40
−20
0
20
40
60
80
Latitude (°N)
16
12
8
4
0
−4
−8
−12
−16
−20
Sv

CHAPTER 3

Model Configurations

Output from COSIMA-based ocean-sea ice models.

MOM-SIS 0.25° Diagnostics

This notebook calculates and retains key diagnostics from our mom025 simulations.

The following experiments are included:

Experiment Name	Description
mom025_nyf	Original simulation, rerun from WOA13 initial conditions.
mom025_nyf_salt	New salt restoring file

Last updated May 20 2017.

Experiments

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
from cosima_cookbook import build_index, expts

# select only mom025 experiments
expts = [expt for expt in expts if 'mom025' in expt]
```

```
expts
```

```
['mom025/mom025_jra_ryf0304',
 'mom025/mom025_jra_ryf8485',
 'mom025/mom025_jra_ryf9091',
 'mom025/mom025_nyf',
 'mom025/mom025_nyf_salt']
```

```
DataDir = '/g/data3/hh5/tmp/cosima/mom025'
expts = ['mom025_nyf']
```

MOM-SIS 0.1° Diagnostics

Key diagnostics from our mom01v5 simulations.

The following experiments are included:

Experiment Name	Description
GFDL50	Original simulation with 50 vertical levels. Ran from Levitus for about 60 years, but data output only saved from about year 40.
KDS75	Branched from GFDL50 at year 45 (re-zeroed), but with Kial Stewart's 75 level scheme. Has now run for 103 years. Years 90-100 have 5-daily output.
KDS75_wind	Short (5-year) Antarctic wind perturbation case, branched from KDS75 at year 40.
KDS75_PI	Paul Spence's Poleward Intensification wind experiment. Branched from KDS75 at year 70, will run until year 100 with 5-daily output for the last decade
KDS75_UP	Paul Spence's Increased winds case. Branched from KDS75 at year 70, will run until year 100 with 5-daily output for the last decade. (In Progress)

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

CHAPTER 4

Contributed Notebooks

Ice Validation

Author: Adele Morrison

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
from glob import glob
import os,sys
import matplotlib.pyplot as plt
import numpy as np
import netCDF4 as nc
from mpl_toolkits.basemap import Basemap
```

```
# model output here:
model = 'mom025'
DataDir = '/g/data3/hh5/tmp/cosima/'+model+'/'
expt = 'mom025_nyf'
expdir = os.path.join(DataDir, expt)
# sea ice observation data here:
ObsDir = '/g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/'
ObsDirExt = '/g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/'

figdir = '/g/data/v45/akm157/figures/mom025_diagnostics/sea_ice/'

# get model grid data:
gridFileList = glob(os.path.join(expdir, 'output*/ocean_grid.nc'))
gridFileList.sort()
ncFile = nc.Dataset(gridFileList[0])
xt_ocean = ncFile.variables['xt_ocean'][...]
yt_ocean = ncFile.variables['yt_ocean'][...]
lon_t = ncFile.variables['geolon_t'][...]
```

```
lat_t = ncFile.variables['geolat_t'][...]
area_t = ncFile.variables['area_t'][...]
NLAT_half = int(np.shape(area_t)[0]/2)
ht = ncFile.variables['ht'][...]
land_mask = np.copy(ht)
land_mask[np.where(ht>30)] = 0
land_mask[np.where(ht<30)] = 1

# path to model sea ice output:
dataFileList = glob(os.path.join(expdir, 'output*/ice_month.nc'))
dataFileList.sort()

# paths to obs output:
obsNHFileList = glob(os.path.join(ObsDir, 'nh/*.nc'))
obsNHFileList.sort()
obsSHFileList = glob(os.path.join(ObsDir, 'sh/*.nc'))
obsSHFileList.sort()
obsExtSHFileList = glob(os.path.join(ObsDirExt, 'sh/*.csv'))
obsExtSHFileList.sort()
obsExtNHFileList = glob(os.path.join(ObsDirExt, 'nh/*.csv'))
obsExtNHFileList.sort()

font = {'size':13}
tick_font=13
```

```
##########################################
# time series of north and south annual sea ice volume
NH_ice_volume = []
SH_ice_volume = []
time = []
for IceFile in dataFileList:
    print('opening '+IceFile)
    ncFile = nc.Dataset(IceFile)
    HI = ncFile.variables['HI'][...]
    time_temp = ncFile.variables['time'][...]
    volume_temp = np.sum(HI*area_t,axis=-1)
    NH_ice_volume = np.append(NH_ice_volume,np.sum(volume_temp[:,NLAT_half:],axis=-1))
    SH_ice_volume = np.append(SH_ice_volume,np.sum(volume_temp[:,:NLAT_half],axis=-1))
    time = np.append(time,time_temp)

# do annual averages:
n_years = int(len(time)/12)
NH_annual_volume = np.zeros(n_years)
SH_annual_volume = np.zeros(n_years)
time_annual = np.zeros(n_years)
for year in range(n_years):
    NH_annual_volume[year] = np.mean(NH_ice_volume[year*12:(year+1)*12])
    SH_annual_volume[year] = np.mean(SH_ice_volume[year*12:(year+1)*12])
    time_annual[year] = np.mean(time[year*12:(year+1)*12])

plt.figure(1,(12,5))
plt.clf()
plt.subplot(1,2,1)
plt.plot(time_annual/365+1,NH_annual_volume/1e12)
plt.xlabel('Year',font)
plt.ylabel(r'ice volume (10$^{12}$ m$^3$)',font)
plt.title('Arctic ice volume',font)
plt.ylim((17.5,20.5))
```

```
plt.subplot(1,2,2)
plt.plot(time_annual/365+1,SH_annual_volume/1e12)
plt.xlabel('Year',font)
plt.ylabel(r'ice volume (10$^{12}$ m$^3$)',font)
plt.title('Antarctic ice volume',font)
plt.ylim((7,8.5))
plt.show()
plt.draw()

plt.show()
```

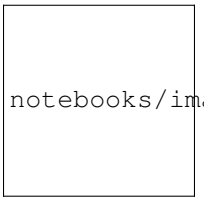
```
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output000/ice_month.nc
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/numpy/ma/core.
↪py:1016: RuntimeWarning: overflow encountered in multiply
  result = self.f(da, db, *args, **kwargs)
```

```
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output001/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output002/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output003/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output004/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output005/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output006/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output007/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output008/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output009/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output010/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output011/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output012/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output013/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output014/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output015/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output016/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output017/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output018/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output019/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output020/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output021/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output022/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output023/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output024/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output025/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output026/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output027/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output028/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output029/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output030/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output031/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output032/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output033/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output034/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output035/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output036/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output037/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output038/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output039/ice_month.nc
```

```
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output040/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output041/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output042/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output043/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output044/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output045/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output046/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output047/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output048/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output049/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output050/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output051/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output052/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output053/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output054/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output055/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output056/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output057/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output058/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output059/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output060/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output061/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output062/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output063/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output064/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output065/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output066/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output067/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output068/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output069/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output070/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output071/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output072/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output073/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output074/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output075/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output076/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output077/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output078/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output079/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output080/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output081/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output082/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output083/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output084/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output085/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output086/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output087/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output088/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output089/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output090/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output091/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output092/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output093/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output094/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output095/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output096/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output097/ice_month.nc
```

```
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output098/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output099/ice_month.nc
```




```
<matplotlib.figure.Figure at 0x7f2c7392ca58>
```

```
################## extent seasonal cycle
## obs extent:
# just take years 79-98 where available:
SH_extent_climatology_obs = np.zeros(12)
import csv
for month in range(12):
    n_years = 0
    print('opening '+obsExtSHFileList[month])
    with open(obsExtSHFileList[month]) as csvfile:
            reader = csv.reader(csvfile)
            rownum = 0
            for row in reader:
                    if (rownum > 0 and float(row[-2])>0 and float(row[0])>1978 \
                            and float(row[0])<1999):
                            SH_extent_climatology_obs[month] += float(row[-2])
                            n_years = n_years + 1
                    rownum = rownum + 1
    SH_extent_climatology_obs[month] = SH_extent_climatology_obs[month] / n_years

NH_extent_climatology_obs = np.zeros(12)
import csv
for month in range(12):
    n_years = 0
    print('opening '+obsExtNHFileList[month])
    with open(obsExtNHFileList[month]) as csvfile:
            reader = csv.reader(csvfile)
            rownum = 0
            for row in reader:
                    if (rownum > 0 and float(row[-2])>0 and float(row[0])>1978 \
                            and float(row[0])<1999):
                            NH_extent_climatology_obs[month] += float(row[-2])
                            n_years = n_years + 1
                    rownum = rownum + 1
    NH_extent_climatology_obs[month] = NH_extent_climatology_obs[month] / n_years
```

```
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_01_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_02_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_03_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_04_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_05_extent_v2.1.
↪csv
```

```
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_06_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_07_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_08_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_09_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_10_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_11_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/sh/S_12_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_01_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_02_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_03_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_04_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_05_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_06_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_07_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_08_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_09_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_10_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_11_extent_v2.1.
↪csv
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02135_extent_monthly/nh/N_12_extent_v2.1.
↪csv
```

```
# climatological seasonal cycle of annual sea ice area - take monthly averages of
# concentration, then sum area where CN > 15%
# for last 10 years of control:
ice_area_south = []
ice_area_north = []
# length of climatology:
n_years = 10
if model=='mom025':
    n_files = n_years
elif model=='mom01v5':
    n_files = 4*n_years
for IceFile in dataFileList[-n_files:]:
    print('opening '+IceFile)
    ncFile = nc.Dataset(IceFile)
    CN = np.sum(ncFile.variables['CN'][...],axis=1)
    for month in range(CN.shape[0]):
            area_field = np.where(CN[month,...]>0.15,area_t,0)
            ice_area_south = np.append(ice_area_south,np.sum(area_field[:NLAT_half,...
↪]))
```

```
            ice_area_north = np.append(ice_area_north,np.sum(area_field[NLAT_half:,...
↪]))
```

```
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output090/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output091/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output092/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output093/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output094/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output095/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output096/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output097/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output098/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output099/ice_month.nc
```

```
# do annual averages:
ice_area_south_climatology = np.zeros(12)
ice_area_north_climatology = np.zeros(12)
for month in range(12):
    ice_area_south_climatology[month] = np.mean(ice_area_south[month::12])
    ice_area_north_climatology[month] = np.mean(ice_area_north[month::12])
# sort months from Jan -> Dec (for mom01 we don't necessarily start with Jan data)
time = ncFile.variables['time']
time_convert = nc.num2date(time[-1],time.units,time.calendar)
last_month = time_convert.month
last_year = time_convert.year
ice_area_south_climatology = np.roll(ice_area_south_climatology,last_month)
ice_area_north_climatology = np.roll(ice_area_north_climatology,last_month)

# Arctic seasonal cycle:
plt.figure(2,(12,5))
plt.clf()
plt.subplot(1,2,1)
plt.plot(np.arange(12)+1,ice_area_north_climatology/1e12,label = 'model')
plt.plot(np.arange(12)+1,NH_extent_climatology_obs)
plt.xlabel('Month',font)
plt.ylabel(r'Sea ice extent (10$^{12}$m$^2$)',font)
plt.title('Arctic extent, yrs '+str(last_year-(n_years-1))+'-'+str(last_year),font)
plt.ylim((0,20))
plt.xlim((1,12))

# Antarctic seasonal cycle:
plt.subplot(1,2,2)
plt.plot(np.arange(12)+1,ice_area_south_climatology/1e12,label = 'model')
plt.plot(np.arange(12)+1,SH_extent_climatology_obs,label="obs 1979-1998")
plt.xlabel('Month',font)
plt.ylabel(r'Sea ice extent (10$^{12}$m$^2$)',font)
plt.ylim((0,25))
plt.xlim((1,12))
plt.title('Antarctic extent, yrs '+str(last_year-(n_years-1))+'-'+str(last_year),font)
plt.legend(prop=font,loc=2)

plt.show()
```

notebooks/images/IceValidationmom025_1.png

```
##########################################
# maps of summer/winter concentration:
####### obs climatology:
# for obs just use years 1988-1997, because concentration not available before then␣
↪and
# climate change after then
# Arctic March:
CN_obs_Mar_NH = 0
# length of climatology:
obs_Mar_list = [IceFile for IceFile in obsNHFileList if (IceFile.find('03_v02')>0 and
    IceFile.find('nh_f')>0 and float(IceFile[-16:-12])>1987 and \
    float(IceFile[-16:-12])<1998)]
n_years = len(obs_Mar_list)
for IceFile in obs_Mar_list:
    print('opening '+IceFile)
    ncFile = nc.Dataset(IceFile)
    CN_obs_Mar_NH = CN_obs_Mar_NH + ncFile.variables['seaice_conc_monthly_cdr'][0,...]
obs_lat_NH = ncFile.variables['latitude'][...]
obs_lon_NH = ncFile.variables['longitude'][...]
# divide by n_years and mask land / arctic pole hole:
CN_obs_Mar_NH = np.ma.masked_where(CN_obs_Mar_NH<0,CN_obs_Mar_NH) / n_years
CN_obs_Mar_NH = np.ma.masked_where(CN_obs_Mar_NH>2,CN_obs_Mar_NH)
```

```
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f08_198803_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f08_198903_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f08_199003_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f08_199103_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f11_199203_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f11_199303_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f11_199403_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f11_199503_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f13_199603_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f13_199703_v02r00.nc
```

```
# Arctic September:
CN_obs_Sep_NH = 0
# length of climatology:
obs_Sep_list = [IceFile for IceFile in obsNHFileList if (IceFile.find('09_v02')>0 and
    IceFile.find('nh_f')>0 and float(IceFile[-16:-12])>1987 and \
    float(IceFile[-16:-12])<1998)]
```

```
n_years = len(obs_Sep_list)
for IceFile in obs_Sep_list:
    print('opening '+IceFile)
    ncFile = nc.Dataset(IceFile)
    CN_obs_Sep_NH = CN_obs_Sep_NH + ncFile.variables['seaice_conc_monthly_cdr'][0,...]
# divide by n_years and mask land / arctic pole hole:
CN_obs_Sep_NH = np.ma.masked_where(CN_obs_Sep_NH<0,CN_obs_Sep_NH) / n_years
CN_obs_Sep_NH = np.ma.masked_where(CN_obs_Sep_NH>2,CN_obs_Sep_NH)

# Antarctic September:
CN_obs_Sep_SH = 0
# length of climatology:
obs_Sep_list = [IceFile for IceFile in obsSHFileList if (IceFile.find('09_v02')>0 and
    IceFile.find('sh_f')>0 and float(IceFile[-16:-12])>1987 and \
    float(IceFile[-16:-12])<1998)]
n_years = len(obs_Sep_list)
for IceFile in obs_Sep_list:
    print('opening '+IceFile)
    ncFile = nc.Dataset(IceFile)
    CN_obs_Sep_SH = CN_obs_Sep_SH + ncFile.variables['seaice_conc_monthly_cdr'][0,...]
obs_lat_SH = ncFile.variables['latitude'][...]
obs_lon_SH = ncFile.variables['longitude'][...]
# divide by n_years and mask land / arctic pole hole:
CN_obs_Sep_SH = np.ma.masked_where(CN_obs_Sep_SH<0,CN_obs_Sep_SH) / n_years
CN_obs_Sep_SH = np.ma.masked_where(CN_obs_Sep_SH>2,CN_obs_Sep_SH)

# Antarctic February:
CN_obs_Feb_SH = 0
# length of climatology:
obs_Feb_list = [IceFile for IceFile in obsSHFileList if (IceFile.find('02_v02')>0 and
    IceFile.find('sh_f')>0 and float(IceFile[-16:-12])>1987 and \
    float(IceFile[-16:-12])<1998)]
n_years = len(obs_Feb_list)
for IceFile in obs_Feb_list:
    print('opening '+IceFile)
    ncFile = nc.Dataset(IceFile)
    CN_obs_Feb_SH = CN_obs_Feb_SH + ncFile.variables['seaice_conc_monthly_cdr'][0,...]
# divide by n_years and mask land / arctic pole hole:
CN_obs_Feb_SH = np.ma.masked_where(CN_obs_Feb_SH<0,CN_obs_Feb_SH) / n_years
CN_obs_Feb_SH = np.ma.masked_where(CN_obs_Feb_SH>2,CN_obs_Feb_SH)

# maps of time mean Antarctic September and Feb (Arctic September and March) sea ice
# concentration with 50% contour for obs and model drawn on

CN_Feb = 0
CN_Mar = 0
CN_Sep = 0
# length of climatology:
n_years = 10
if model=='mom025':
    n_files = n_years
elif model=='mom01v5':
    n_files = 4*n_years
for IceFile in dataFileList[-n_files:]:
    print('opening '+IceFile)
    ncFile = nc.Dataset(IceFile)
    time = ncFile.variables['time']
    for month_n in range(len(time)):
```

```
            # check calendar month:
            time_convert = nc.num2date(time[month_n],time.units,time.calendar)
            month = time_convert.month
            # sum over concentration levels:
            if month == 2:
                    CN_Feb = CN_Feb + np.sum(ncFile.variables['CN'][month_n,:,:NLAT_
↪half,:],axis=0)
            elif month == 3:
                    CN_Mar = CN_Mar + np.sum(ncFile.variables['CN'][month_n,:,NLAT_
↪half:,:],axis=0)
            elif month == 9:
                    CN_Sep = CN_Sep + np.sum(ncFile.variables['CN'][month_n,:,:,:],
↪axis=0)
# divide by n_years and mask:
CN_Feb = np.ma.masked_where(land_mask[:NLAT_half,:]==1,CN_Feb / n_years)
CN_Mar = np.ma.masked_where(land_mask[NLAT_half:,:]==1,CN_Mar / n_years)
CN_Sep = np.ma.masked_where(land_mask==1,CN_Sep / n_years)

last_year = time_convert.year

levels = np.arange(0,1.01,.01)
font = {'size':13}
tick_font=13
```

```
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f08_198809_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f08_198909_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f08_199009_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f08_199109_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f11_199209_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f11_199309_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f11_199409_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f11_199509_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f13_199609_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/nh/seaice_conc_
↪monthly_nh_f13_199709_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f08_198809_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f08_198909_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f08_199009_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f08_199109_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f11_199209_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f11_199309_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f11_199409_v02r00.nc
```

```
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f11_199509_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f13_199609_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f13_199709_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f08_198802_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f08_198902_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f08_199002_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f08_199102_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f11_199202_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f11_199302_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f11_199402_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f11_199502_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f13_199602_v02r00.nc
opening /g/data/v45/akm157/data/NSIDC/NOAA_G02202_v2_conc_monthly/sh/seaice_conc_
↪monthly_sh_f13_199702_v02r00.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output090/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output091/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output092/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output093/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output094/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output095/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output096/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output097/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output098/ice_month.nc
opening /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output099/ice_month.nc
```

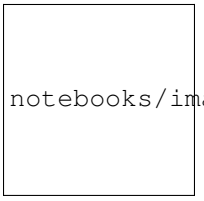
```
# Antarctic concentration maps:
plt.figure(3,(12,7))
plt.clf()
plt.subplot(1,2,1)
m = Basemap(projection ='spstere',boundinglat=-50,lon_0=180,resolution='l',round='True
↪')
x,y = m(*(lon_t[:NLAT_half,:],lat_t[:NLAT_half,:]))
xobs,yobs = m(*(obs_lon_SH,obs_lat_SH))
ctr = m.contourf(x,y,CN_Feb,levels=levels,cmap='viridis')
plt.title('February min, yrs '+str(last_year-(n_years-1))+'-'+
    str(last_year),font,y=1.03)
cbar = m.colorbar(ctr, location = 'bottom', pad = "6%")
cbar.set_label('Ice concentration',size=tick_font)
cbar.set_ticks([0,.2,.4,.6,.8,1])
cbar_labels=plt.getp(cbar.ax.axes,'xticklabels')
plt.setp(cbar_labels,fontsize=tick_font)
#m.contour(x,y,land_mask[:NLAT_half,:],[0,1],colors='w')
m.contour(x,y,CN_Feb,[0.3],colors='w')
# obs:
m.contour(xobs,yobs,CN_obs_Feb_SH,[0.3],colors='r')
```

```
plt.subplot(1,2,2)
m = Basemap(projection ='spstere',boundinglat=-50,lon_0=180,resolution='l',round='True
↪')
ctr = m.contourf(x,y,CN_Sep[:NLAT_half,:],levels=levels,cmap='viridis')
plt.title('September max, yrs '+str(last_year-(n_years-1))+'-'+
    str(last_year),font,y=1.03)
cbar = m.colorbar(ctr, location = 'bottom', pad = "6%")
cbar.set_label('Ice concentration',size=tick_font)
cbar.set_ticks([0,.2,.4,.6,.8,1])
cbar_labels=plt.getp(cbar.ax.axes,'xticklabels')
plt.setp(cbar_labels,fontsize=tick_font)
#m.contour(x,y,land_mask[:NLAT_half,:],[0,1],colors='w')
m.contour(x,y,CN_Sep[:NLAT_half,:],[0.3],colors='w')
# obs:
m.contour(xobs,yobs,CN_obs_Sep_SH,[0.3],colors='r')
plt.show()
```

notebooks/images/IceValidationmom025_2.png

```
# Arctic concentration maps:
plt.figure(4,(12,7))
plt.clf()
plt.subplot(1,2,1)
m = Basemap(projection ='npstere',boundinglat=48,lon_0=0,resolution='l')
x,y = m(*(lon_t[NLAT_half:,:],lat_t[NLAT_half:,:]))
xobs,yobs = m(*(obs_lon_NH,obs_lat_NH))
ctr = m.contourf(x,y,CN_Sep[NLAT_half:,:],levels=levels,cmap='viridis')
plt.title('September min, yrs '+str(last_year-(n_years-1))+'-'+
    str(last_year),font,y=1.03)
cbar = m.colorbar(ctr, location = 'bottom', pad = "6%")
cbar.set_label('Ice concentration',size=tick_font)
cbar.set_ticks([0,.2,.4,.6,.8,1])
cbar_labels=plt.getp(cbar.ax.axes,'xticklabels')
plt.setp(cbar_labels,fontsize=tick_font)
#m.contour(x,y,land_mask[NLAT_half:,:],[0,1],colors='w')
m.contour(x,y,CN_Sep[NLAT_half:,:],[0.3],colors='w')
# obs:
m.contour(xobs,yobs,CN_obs_Sep_NH,[0.3],colors='r')

plt.subplot(1,2,2)
m = Basemap(projection ='npstere',boundinglat=48,lon_0=0,resolution='l')
ctr = m.contourf(x,y,CN_Mar,levels=levels,cmap='viridis')
plt.title('March max, yrs '+str(last_year-(n_years-1))+'-'+
    str(last_year),font,y=1.03)
cbar = m.colorbar(ctr, location = 'bottom', pad = "6%")
cbar.set_label('Ice concentration',size=tick_font)
cbar.set_ticks([0,.2,.4,.6,.8,1])
cbar_labels=plt.getp(cbar.ax.axes,'xticklabels')
plt.setp(cbar_labels,fontsize=tick_font)
#m.contour(x,y,land_mask[NLAT_half:,:],[0,1],colors='w')
m.contour(x,y,CN_Mar,[0.3],colors='w')
# obs:
```

```
m.contour(xobs,yobs,CN_obs_Mar_NH,[0.3],colors='r')

plt.show()
```



MOM Run Summary

MOM runs can be identified by the presence of key files such as `diag_table` and `data_table`

Goal: generate a class that represents MOM metadata and a JSON representation

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
import os
import tqdm
import yaml
import f90nml
import csv
import datetime
```

```
directoriesToSearch = ['/g/data3/hh5/tmp/cosima',
                       ]
```

```
output_dirs = []
for projectDir in directoriesToSearch:
    for dirpath, dirnames, filenames in os.walk(projectDir):
        if 'diag_table' in filenames:
            output_dirs.append(dirpath)
```

File| Description —- | ———- input.nml | Principal configuration diag_table | Diagnostic output management data_table | Input and boundary condition data field management field_table | Initial condition and advection scheme configuration config.yaml | logfile.000000.out | mom.err | mom.out | time_stamp.out |

Output files: ———- | ice_month.nc | ocean__0096_184.nc|

ocean__0096_189.nc ocean__0096_194.nc ocean__0096_199.nc ocean__0096_204.nc ocean__0096_209.nc ocean__0096_214.nc ocean__0096_219.nc ocean__0096_224.nc ocean__0096_229.nc ocean__0096_234.nc ocean__0096_239.nc ocean__0096_244.nc ocean__0096_249.nc ocean__0096_254.nc ocean__0096_259.nc ocean__0096_264.nc ocean__0096_269.nc ocean_grid.nc ocean_month.nc ocean.nc ocean_scalar.nc

```
class Run():
    def __init__(self, dirpath):
        self.config = yaml.load(open(os.path.join(dirpath, 'config.yaml')))

        self.input = f90nml.read(os.path.join(dirpath, 'input.nml'))

        # diag_table
```

```
        # http://data1.gfdl.noaa.gov/~nnz/MOM/mom5_pubrel_August2012/src/shared/diag_
↪manager/diag_table.html
        with open(os.path.join(dirpath, 'diag_table'), 'r') as f:
            reader = csv.reader(f)
            self.diag_table = []
            for row in reader:
                if len(row) > 0 and row[0][0] == '#':
                    continue
                self.diag_table.append([value.replace('"', '').strip() for value in␣
↪row])

        # Global Section
        self.title = self.diag_table[0][0] # first line
        self.base_date = self.diag_table[1][0]

        # 1 1 1 0 0 0  => 0001 1 1 0 0 0 for a valid datetime string
        if self.base_date.split()[0] == '1':
            self.base_date = '000' + self.base_date

        # year month day hour minute second
        self.base_date = datetime.datetime.strptime(self.base_date, '%Y %m %d %H %M %S
↪')

        self.files = {}
        self.fields = {}
        # Files and Fields can be mixed
        for line in self.diag_table[2:]:
            if len(line) == 0:
                continue

            try:
                int(line[1]) # Files have integers in field 2
            except:
                file_line = False
            else:
                file_line = True

            if file_line:
                # File Section
                self.files[line[0]] = line
            else:
                # Field Section
                print(line)
                self.fields[line[1]] = line

        with open(os.path.join(dirpath, 'data_table'), 'r') as f:
            reader = csv.reader(f)
            self.data_table = []
            for row in reader:
                if row[0][0] == '#':
                    continue
                self.data_table.append([value.replace('"', '').strip() for value in␣
↪row])

        with open(os.path.join(dirpath, 'field_table'), 'r') as f:
            reader = csv.reader(f)
            self.field_table = []
            for row in reader:
```

```
                if len(row) > 0 and row[0][0] == '#':
                    continue
                self.field_table.append([value.replace('"', '').strip() for value in␣
↪row])

        lines = open(os.path.join(dirpath, 'time_stamp.out'), 'r').readlines()
        self.time_stamp = [line.strip() for line in lines]
```

```
run = Run(output_dirs[0])
```

```
['ocean_model', 'geolon_t', 'geolon_t', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'geolat_t', 'geolat_t', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'geolon_c', 'geolon_c', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'geolat_c', 'geolat_c', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'ht', 'ht', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'hu', 'hu', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'area_t', 'area_t', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'area_u', 'area_u', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'kmt', 'kmt', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'kmu', 'kmu', 'ocean_grid', 'all', '.false.', 'none', '2']
['ocean_model', 'drag_coeff', 'drag_coeff', 'ocean_grid', 'all', '.false.', 'none', '2
↪']
['ocean_model', 'geolon_t', 'geolon_t', 'ocean', 'all', '.false.', 'none', '2']
['ocean_model', 'geolat_t', 'geolat_t', 'ocean', 'all', '.false.', 'none', '2']
['ocean_model', 'geolon_c', 'geolon_c', 'ocean', 'all', '.false.', 'none', '2']
['ocean_model', 'geolat_c', 'geolat_c', 'ocean', 'all', '.false.', 'none', '2']
['ocean_model', 'age_global', 'age_global', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'temp', 'temp', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'salt', 'salt', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'u', 'u', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'v', 'v', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'pot_rho_0', 'pot_rho_0', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'ty_trans_rho', 'ty_trans_rho', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'ty_trans', 'ty_trans', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'tx_trans', 'tx_trans', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'wt', 'wt', 'ocean', 'all', '.true.', 'none', '2']
['ocean_model', 'geolon_t', 'geolon_t', 'ocean_month', 'all', '.false.', 'none', '2']
['ocean_model', 'geolat_t', 'geolat_t', 'ocean_month', 'all', '.false.', 'none', '2']
['ocean_model', 'geolon_c', 'geolon_c', 'ocean_month', 'all', '.false.', 'none', '2']
['ocean_model', 'geolat_c', 'geolat_c', 'ocean_month', 'all', '.false.', 'none', '2']
['ocean_model', 'mld', 'mld', 'ocean_month', 'all', '.true.', 'none', '2']
['ocean_model', 'net_sfc_heating', 'net_sfc_heating', 'ocean_month', 'all', '.true.',
↪'none', '2']
['ocean_model', 'tx_trans_int_z', 'tx_trans_int_z', 'ocean_month', 'all', '.true.',
↪'none', '2']
['ocean_model', 'ty_trans_int_z', 'ty_trans_int_z', 'ocean_month', 'all', '.true.',
↪'none', '2']
['ocean_model', 'temp_xflux_adv_int_z', 'temp_xflux_adv_int_z', 'ocean_month', 'all',
↪'.true.', 'none', '2']
['ocean_model', 'temp_yflux_adv_int_z', 'temp_yflux_adv_int_z', 'ocean_month', 'all',
↪'.true.', 'none', '2']
['ocean_model', 'temp_xflux_submeso_int_z', 'temp_xflux_submeso_int_z', 'ocean_month',
↪ 'all', '.true.', 'none', '2']
['ocean_model', 'temp_yflux_submeso_int_z', 'temp_yflux_submeso_int_z', 'ocean_month',
↪ 'all', '.true.', 'none', '2']
['ocean_model', 'tau_x', 'tau_x', 'ocean_month', 'all', '.true.', 'none', '2']
['ocean_model', 'tau_y', 'tau_y', 'ocean_month', 'all', '.true.', 'none', '2']
```

```
['ice_model', 'HI', 'HI', 'ice_month', 'all', '.true.', 'none', '2']
['ice_model', 'HS', 'HS', 'ice_month', 'all', '.true.', 'none', '2']
['ice_model', 'CN', 'CN', 'ice_month', 'all', '.true.', 'none', '2']
['ocean_model', 'geolon_t', 'geolon_t', 'oceankerg_%4yr_%3dy', 'all', '.false.', '-
↪280,80,-81,-30,-1,-1', '2']
['ocean_model', 'geolat_t', 'geolat_t', 'oceankerg_%4yr_%3dy', 'all', '.false.', '-
↪280,80,-81,-30,-1,-1', '2']
['ocean_model', 'geolon_c', 'geolon_c', 'oceankerg_%4yr_%3dy', 'all', '.false.', '-
↪280,80,-81,-30,-1,-1', '2']
['ocean_model', 'geolat_c', 'geolat_c', 'oceankerg_%4yr_%3dy', 'all', '.false.', '-
↪280,80,-81,-30,-1,-1', '2']
['ocean_model', 'u', 'u', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-280,80,-81,-30,-1,
↪-1', '2']
['ocean_model', 'v', 'v', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-280,80,-81,-30,-1,
↪-1', '2']
['ocean_model', 'wt', 'wt', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-280,80,-81,-30,-
↪1,-1', '2']
['ocean_model', 'temp', 'temp', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-280,80,-81,-
↪30,-1,-1', '2']
['ocean_model', 'salt', 'salt', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-280,80,-81,-
↪30,-1,-1', '2']
['ocean_model', 'pot_rho_0', 'pot_rho_0', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-
↪280,80,-81,-30,-1,-1', '2']
['ocean_model', 'mld', 'mld', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-280,80,-81,-
↪30,-1,-1', '2']
['ocean_model', 'tau_x', 'tau_x', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-280,80,-
↪81,-30,-1,-1', '2']
['ocean_model', 'tau_y', 'tau_y', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-280,80,-
↪81,-30,-1,-1', '2']
['ocean_model', 'sea_level', 'sea_level', 'oceankerg_%4yr_%3dy', 'all', '.true.', '-
↪280,80,-81,-30,-1,-1', '2']
['ocean_model', 'sea_level_sq', 'sea_level_sq', 'oceankerg_%4yr_%3dy', 'all', '.true.
↪', '-280,80,-81,-30,-1,-1', '2']
['ocean_model', 'ke_tot', 'ke_tot', 'ocean_scalar', 'all', '.true.', 'none', '2']
['ocean_model', 'pe_tot', 'pe_tot', 'ocean_scalar', 'all', '.true.', 'none', '2']
['ocean_model', 'temp_global_ave', 'temp_global_ave', 'ocean_scalar', 'all', '.true.',
↪ 'none', '2']
['ocean_model', 'salt_global_ave', 'salt_global_ave', 'ocean_scalar', 'all', '.true.',
↪ 'none', '2']
['ocean_model', 'rhoave', 'rhoave', 'ocean_scalar', 'all', '.true.', 'none', '2']
['ocean_model', 'temp_surface_ave', 'temp_surface_ave', 'ocean_scalar', 'all', '.true.
↪', 'none', '2']
['ocean_model', 'salt_surface_ave', 'salt_surface_ave', 'ocean_scalar', 'all', '.true.
↪', 'none', '2']
['ocean_model', 'total_ocean_salt', 'total_ocean_salt', 'ocean_scalar', 'all', '.true.
↪', 'none', '2']
['ocean_model', 'total_ocean_heat', 'total_ocean_heat', 'ocean_scalar', 'all', '.true.
↪', 'none', '2']
['ocean_model', 'eta_global', 'eta_global', 'ocean_scalar', 'all', '.true.', 'none',
↪'2']
['ocean_model', 'total_volume_seawater', 'total_volume_seawater', 'ocean_scalar', 'all
↪', '.true.', 'none', '2']
['ocean_model', 'total_ocean_sfc_salt_flux_coupler', 'total_ocean_sfc_salt_flux_
↪coupler', 'ocean_scalar', 'all', '.true.', 'none', '2']
['ocean_model', 'total_net_sfc_heating', 'total_net_sfc_heating', 'ocean_scalar', 'all
↪', '.true.', 'none', '2']
['ocean_model', 'total_ocean_pme_river', 'total_ocean_pme_river', 'ocean_scalar', 'all
↪', '.true.', 'none', '2']
```

```
['ocean_model', 'total_ocean_river', 'total_ocean_river', 'ocean_scalar', 'all', '.
↪true.', 'none', '2']
['ocean_model', 'total_ocean_runoff', 'total_ocean_runoff', 'ocean_scalar', 'all', '.
↪true.', 'none', '2']
['ocean_model', 'total_ocean_calving', 'total_ocean_calving', 'ocean_scalar', 'all',
↪'.true.', 'none', '2']
['ocean_model', 'total_ocean_melt', 'total_ocean_melt', 'ocean_scalar', 'all', '.true.
↪', 'none', '2']
['ocean_model', 'total_ocean_evap', 'total_ocean_evap', 'ocean_scalar', 'all', '.true.
↪', 'none', '2']
['ocean_model', 'total_ocean_lprec', 'total_ocean_lprec', 'ocean_scalar', 'all', '.
↪true.', 'none', '2']
['ocean_model', 'total_ocean_fprec', 'total_ocean_fprec', 'ocean_scalar', 'all', '.
↪true.', 'none', '2']
['ocean_model', 'total_ocean_runoff_heat', 'total_ocean_runoff_heat', 'ocean_scalar',
↪'all', '.true.', 'none', '2']
['ocean_model', 'total_ocean_calving_heat', 'total_ocean_calving_heat', 'ocean_scalar
↪', 'all', '.true.', 'none', '2']
['ocean_model', 'total_ocean_river_heat', 'total_ocean_river_heat', 'ocean_scalar',
↪'all', '.true.', 'none', '2']
['ocean_model', 'total_ocean_hflux_prec', 'total_ocean_hflux_prec', 'ocean_scalar',
↪'all', '.true.', 'none', '2']
['ocean_model', 'total_ocean_hflux_evap', 'total_ocean_hflux_evap', 'ocean_scalar',
↪'all', '.true.', 'none', '2']
['ocean_model', 'total_ocean_hflux_coupler', 'total_ocean_hflux_coupler', 'ocean_
↪scalar', 'all', '.true.', 'none', '2']
['ocean_model', 'total_ocean_swflx', 'total_ocean_swflx', 'ocean_scalar', 'all', '.
↪true.', 'none', '2']
['ocean_model', 'total_ocean_swflx_vis', 'total_ocean_swflx_vis', 'ocean_scalar', 'all
↪', '.true.', 'none', '2']
['ocean_model', 'total_ocean_lw_heat', 'total_ocean_lw_heat', 'ocean_scalar', 'all',
↪'.true.', 'none', '2']
['ocean_model', 'total_ocean_evap_heat', 'total_ocean_evap_heat', 'ocean_scalar', 'all
↪', '.true.', 'none', '2']
['ocean_model', 'total_ocean_fprec_melt_heat', 'total_ocean_fprec_melt_heat', 'ocean_
↪scalar', 'all', '.true.', 'none', '2']
['ocean_model', 'total_ocean_calving_melt_heat', 'total_ocean_calving_melt_heat',
↪'ocean_scalar', 'all', '.true.', 'none', '2']
['ocean_model', 'total_ocean_sens_heat', 'total_ocean_sens_heat', 'ocean_scalar', 'all
↪', '.true.', 'none', '2']
```

```
run.base_date.isoformat()
```

```
'0001-01-01T00:00:00'
```

```
run.fields
```

```
{'CN': ['ice_model', 'CN', 'CN', 'ice_month', 'all', '.true.', 'none', '2'],
 'HI': ['ice_model', 'HI', 'HI', 'ice_month', 'all', '.true.', 'none', '2'],
 'HS': ['ice_model', 'HS', 'HS', 'ice_month', 'all', '.true.', 'none', '2'],
 'age_global': ['ocean_model',
  'age_global',
  'age_global',
  'ocean',
  'all',
  '.true.',
```

```
 'none',
 '2'],
'area_t': ['ocean_model',
 'area_t',
 'area_t',
 'ocean_grid',
 'all',
 '.false.',
 'none',
 '2'],
'area_u': ['ocean_model',
 'area_u',
 'area_u',
 'ocean_grid',
 'all',
 '.false.',
 'none',
 '2'],
'drag_coeff': ['ocean_model',
 'drag_coeff',
 'drag_coeff',
 'ocean_grid',
 'all',
 '.false.',
 'none',
 '2'],
'eta_global': ['ocean_model',
 'eta_global',
 'eta_global',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'geolat_c': ['ocean_model',
 'geolat_c',
 'geolat_c',
 'oceankerg_%4yr_%3dy',
 'all',
 '.false.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'geolat_t': ['ocean_model',
 'geolat_t',
 'geolat_t',
 'oceankerg_%4yr_%3dy',
 'all',
 '.false.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'geolon_c': ['ocean_model',
 'geolon_c',
 'geolon_c',
 'oceankerg_%4yr_%3dy',
 'all',
 '.false.',
 '-280,80,-81,-30,-1,-1',
 '2'],
```

```
 'geolon_t': ['ocean_model',
  'geolon_t',
  'geolon_t',
  'oceankerg_%4yr_%3dy',
  'all',
  '.false.',
  '-280,80,-81,-30,-1,-1',
  '2'],
 'ht': ['ocean_model',
  'ht',
  'ht',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'hu': ['ocean_model',
  'hu',
  'hu',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'ke_tot': ['ocean_model',
  'ke_tot',
  'ke_tot',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'kmt': ['ocean_model',
  'kmt',
  'kmt',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'kmu': ['ocean_model',
  'kmu',
  'kmu',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'mld': ['ocean_model',
  'mld',
  'mld',
  'oceankerg_%4yr_%3dy',
  'all',
  '.true.',
  '-280,80,-81,-30,-1,-1',
  '2'],
 'net_sfc_heating': ['ocean_model',
  'net_sfc_heating',
```

```
 'net_sfc_heating',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'pe_tot': ['ocean_model',
 'pe_tot',
 'pe_tot',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'pot_rho_0': ['ocean_model',
 'pot_rho_0',
 'pot_rho_0',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'rhoave': ['ocean_model',
 'rhoave',
 'rhoave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'salt': ['ocean_model',
 'salt',
 'salt',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'salt_global_ave': ['ocean_model',
 'salt_global_ave',
 'salt_global_ave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'salt_surface_ave': ['ocean_model',
 'salt_surface_ave',
 'salt_surface_ave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'sea_level': ['ocean_model',
 'sea_level',
 'sea_level',
 'oceankerg_%4yr_%3dy',
```

```
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'sea_level_sq': ['ocean_model',
 'sea_level_sq',
 'sea_level_sq',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'tau_x': ['ocean_model',
 'tau_x',
 'tau_x',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'tau_y': ['ocean_model',
 'tau_y',
 'tau_y',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'temp': ['ocean_model',
 'temp',
 'temp',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'temp_global_ave': ['ocean_model',
 'temp_global_ave',
 'temp_global_ave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'temp_surface_ave': ['ocean_model',
 'temp_surface_ave',
 'temp_surface_ave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'temp_xflux_adv_int_z': ['ocean_model',
 'temp_xflux_adv_int_z',
 'temp_xflux_adv_int_z',
 'ocean_month',
 'all',
 '.true.',
```

```
 'none',
 '2'],
'temp_xflux_submeso_int_z': ['ocean_model',
 'temp_xflux_submeso_int_z',
 'temp_xflux_submeso_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'temp_yflux_adv_int_z': ['ocean_model',
 'temp_yflux_adv_int_z',
 'temp_yflux_adv_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'temp_yflux_submeso_int_z': ['ocean_model',
 'temp_yflux_submeso_int_z',
 'temp_yflux_submeso_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'total_net_sfc_heating': ['ocean_model',
 'total_net_sfc_heating',
 'total_net_sfc_heating',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_calving': ['ocean_model',
 'total_ocean_calving',
 'total_ocean_calving',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_calving_heat': ['ocean_model',
 'total_ocean_calving_heat',
 'total_ocean_calving_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_calving_melt_heat': ['ocean_model',
 'total_ocean_calving_melt_heat',
 'total_ocean_calving_melt_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
```

```
 'total_ocean_evap': ['ocean_model',
  'total_ocean_evap',
  'total_ocean_evap',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_evap_heat': ['ocean_model',
  'total_ocean_evap_heat',
  'total_ocean_evap_heat',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_fprec': ['ocean_model',
  'total_ocean_fprec',
  'total_ocean_fprec',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_fprec_melt_heat': ['ocean_model',
  'total_ocean_fprec_melt_heat',
  'total_ocean_fprec_melt_heat',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_heat': ['ocean_model',
  'total_ocean_heat',
  'total_ocean_heat',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_hflux_coupler': ['ocean_model',
  'total_ocean_hflux_coupler',
  'total_ocean_hflux_coupler',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_hflux_evap': ['ocean_model',
  'total_ocean_hflux_evap',
  'total_ocean_hflux_evap',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_hflux_prec': ['ocean_model',
  'total_ocean_hflux_prec',
```

```
 'total_ocean_hflux_prec',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_lprec': ['ocean_model',
 'total_ocean_lprec',
 'total_ocean_lprec',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_lw_heat': ['ocean_model',
 'total_ocean_lw_heat',
 'total_ocean_lw_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_melt': ['ocean_model',
 'total_ocean_melt',
 'total_ocean_melt',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_pme_river': ['ocean_model',
 'total_ocean_pme_river',
 'total_ocean_pme_river',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_river': ['ocean_model',
 'total_ocean_river',
 'total_ocean_river',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_river_heat': ['ocean_model',
 'total_ocean_river_heat',
 'total_ocean_river_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_runoff': ['ocean_model',
 'total_ocean_runoff',
 'total_ocean_runoff',
 'ocean_scalar',
```

```
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_runoff_heat': ['ocean_model',
 'total_ocean_runoff_heat',
 'total_ocean_runoff_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_salt': ['ocean_model',
 'total_ocean_salt',
 'total_ocean_salt',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_sens_heat': ['ocean_model',
 'total_ocean_sens_heat',
 'total_ocean_sens_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_sfc_salt_flux_coupler': ['ocean_model',
 'total_ocean_sfc_salt_flux_coupler',
 'total_ocean_sfc_salt_flux_coupler',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_swflx': ['ocean_model',
 'total_ocean_swflx',
 'total_ocean_swflx',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_swflx_vis': ['ocean_model',
 'total_ocean_swflx_vis',
 'total_ocean_swflx_vis',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_volume_seawater': ['ocean_model',
 'total_volume_seawater',
 'total_volume_seawater',
 'ocean_scalar',
 'all',
 '.true.',
```

```
 'none',
 '2'],
'tx_trans': ['ocean_model',
 'tx_trans',
 'tx_trans',
 'ocean',
 'all',
 '.true.',
 'none',
 '2'],
'tx_trans_int_z': ['ocean_model',
 'tx_trans_int_z',
 'tx_trans_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'ty_trans': ['ocean_model',
 'ty_trans',
 'ty_trans',
 'ocean',
 'all',
 '.true.',
 'none',
 '2'],
'ty_trans_int_z': ['ocean_model',
 'ty_trans_int_z',
 'ty_trans_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'ty_trans_rho': ['ocean_model',
 'ty_trans_rho',
 'ty_trans_rho',
 'ocean',
 'all',
 '.true.',
 'none',
 '2'],
'u': ['ocean_model',
 'u',
 'u',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'v': ['ocean_model',
 'v',
 'v',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
```

```
 'wt': ['ocean_model',
  'wt',
  'wt',
  'oceankerg_%4yr_%3dy',
  'all',
  '.true.',
  '-280,80,-81,-30,-1,-1',
  '2']}
```

```
output_dirs[0]
```

```
'/g/data3/hh5/tmp/cosima/mom01v5/KDS75_wind/output165'
```

```
run.fields
```

```
{'CN': ['ice_model', 'CN', 'CN', 'ice_month', 'all', '.true.', 'none', '2'],
 'HI': ['ice_model', 'HI', 'HI', 'ice_month', 'all', '.true.', 'none', '2'],
 'HS': ['ice_model', 'HS', 'HS', 'ice_month', 'all', '.true.', 'none', '2'],
 'age_global': ['ocean_model',
  'age_global',
  'age_global',
  'ocean',
  'all',
  '.true.',
  'none',
  '2'],
 'area_t': ['ocean_model',
  'area_t',
  'area_t',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'area_u': ['ocean_model',
  'area_u',
  'area_u',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'drag_coeff': ['ocean_model',
  'drag_coeff',
  'drag_coeff',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'eta_global': ['ocean_model',
  'eta_global',
  'eta_global',
  'ocean_scalar',
  'all',
  '.true.',
```

```
 'none',
 '2'],
'geolat_c': ['ocean_model',
 'geolat_c',
 'geolat_c',
 'oceankerg_%4yr_%3dy',
 'all',
 '.false.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'geolat_t': ['ocean_model',
 'geolat_t',
 'geolat_t',
 'oceankerg_%4yr_%3dy',
 'all',
 '.false.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'geolon_c': ['ocean_model',
 'geolon_c',
 'geolon_c',
 'oceankerg_%4yr_%3dy',
 'all',
 '.false.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'geolon_t': ['ocean_model',
 'geolon_t',
 'geolon_t',
 'oceankerg_%4yr_%3dy',
 'all',
 '.false.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'ht': ['ocean_model',
 'ht',
 'ht',
 'ocean_grid',
 'all',
 '.false.',
 'none',
 '2'],
'hu': ['ocean_model',
 'hu',
 'hu',
 'ocean_grid',
 'all',
 '.false.',
 'none',
 '2'],
'ke_tot': ['ocean_model',
 'ke_tot',
 'ke_tot',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
```

```
 'kmt': ['ocean_model',
  'kmt',
  'kmt',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'kmu': ['ocean_model',
  'kmu',
  'kmu',
  'ocean_grid',
  'all',
  '.false.',
  'none',
  '2'],
 'mld': ['ocean_model',
  'mld',
  'mld',
  'oceankerg_%4yr_%3dy',
  'all',
  '.true.',
  '-280,80,-81,-30,-1,-1',
  '2'],
 'net_sfc_heating': ['ocean_model',
  'net_sfc_heating',
  'net_sfc_heating',
  'ocean_month',
  'all',
  '.true.',
  'none',
  '2'],
 'pe_tot': ['ocean_model',
  'pe_tot',
  'pe_tot',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'pot_rho_0': ['ocean_model',
  'pot_rho_0',
  'pot_rho_0',
  'oceankerg_%4yr_%3dy',
  'all',
  '.true.',
  '-280,80,-81,-30,-1,-1',
  '2'],
 'rhoave': ['ocean_model',
  'rhoave',
  'rhoave',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'salt': ['ocean_model',
  'salt',
```

```
 'salt',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'salt_global_ave': ['ocean_model',
 'salt_global_ave',
 'salt_global_ave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'salt_surface_ave': ['ocean_model',
 'salt_surface_ave',
 'salt_surface_ave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'sea_level': ['ocean_model',
 'sea_level',
 'sea_level',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'sea_level_sq': ['ocean_model',
 'sea_level_sq',
 'sea_level_sq',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'tau_x': ['ocean_model',
 'tau_x',
 'tau_x',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'tau_y': ['ocean_model',
 'tau_y',
 'tau_y',
 'oceankerg_%4yr_%3dy',
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'temp': ['ocean_model',
 'temp',
 'temp',
 'oceankerg_%4yr_%3dy',
```

```
 'all',
 '.true.',
 '-280,80,-81,-30,-1,-1',
 '2'],
'temp_global_ave': ['ocean_model',
 'temp_global_ave',
 'temp_global_ave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'temp_surface_ave': ['ocean_model',
 'temp_surface_ave',
 'temp_surface_ave',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'temp_xflux_adv_int_z': ['ocean_model',
 'temp_xflux_adv_int_z',
 'temp_xflux_adv_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'temp_xflux_submeso_int_z': ['ocean_model',
 'temp_xflux_submeso_int_z',
 'temp_xflux_submeso_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'temp_yflux_adv_int_z': ['ocean_model',
 'temp_yflux_adv_int_z',
 'temp_yflux_adv_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'temp_yflux_submeso_int_z': ['ocean_model',
 'temp_yflux_submeso_int_z',
 'temp_yflux_submeso_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'total_net_sfc_heating': ['ocean_model',
 'total_net_sfc_heating',
 'total_net_sfc_heating',
 'ocean_scalar',
 'all',
 '.true.',
```

```
 'none',
 '2'],
'total_ocean_calving': ['ocean_model',
 'total_ocean_calving',
 'total_ocean_calving',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_calving_heat': ['ocean_model',
 'total_ocean_calving_heat',
 'total_ocean_calving_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_calving_melt_heat': ['ocean_model',
 'total_ocean_calving_melt_heat',
 'total_ocean_calving_melt_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_evap': ['ocean_model',
 'total_ocean_evap',
 'total_ocean_evap',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_evap_heat': ['ocean_model',
 'total_ocean_evap_heat',
 'total_ocean_evap_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_fprec': ['ocean_model',
 'total_ocean_fprec',
 'total_ocean_fprec',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_fprec_melt_heat': ['ocean_model',
 'total_ocean_fprec_melt_heat',
 'total_ocean_fprec_melt_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
```

```
 'total_ocean_heat': ['ocean_model',
  'total_ocean_heat',
  'total_ocean_heat',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_hflux_coupler': ['ocean_model',
  'total_ocean_hflux_coupler',
  'total_ocean_hflux_coupler',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_hflux_evap': ['ocean_model',
  'total_ocean_hflux_evap',
  'total_ocean_hflux_evap',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_hflux_prec': ['ocean_model',
  'total_ocean_hflux_prec',
  'total_ocean_hflux_prec',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_lprec': ['ocean_model',
  'total_ocean_lprec',
  'total_ocean_lprec',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_lw_heat': ['ocean_model',
  'total_ocean_lw_heat',
  'total_ocean_lw_heat',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_melt': ['ocean_model',
  'total_ocean_melt',
  'total_ocean_melt',
  'ocean_scalar',
  'all',
  '.true.',
  'none',
  '2'],
 'total_ocean_pme_river': ['ocean_model',
  'total_ocean_pme_river',
```

```
 'total_ocean_pme_river',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_river': ['ocean_model',
 'total_ocean_river',
 'total_ocean_river',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_river_heat': ['ocean_model',
 'total_ocean_river_heat',
 'total_ocean_river_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_runoff': ['ocean_model',
 'total_ocean_runoff',
 'total_ocean_runoff',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_runoff_heat': ['ocean_model',
 'total_ocean_runoff_heat',
 'total_ocean_runoff_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_salt': ['ocean_model',
 'total_ocean_salt',
 'total_ocean_salt',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_sens_heat': ['ocean_model',
 'total_ocean_sens_heat',
 'total_ocean_sens_heat',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_sfc_salt_flux_coupler': ['ocean_model',
 'total_ocean_sfc_salt_flux_coupler',
 'total_ocean_sfc_salt_flux_coupler',
 'ocean_scalar',
```

```
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_swflx': ['ocean_model',
 'total_ocean_swflx',
 'total_ocean_swflx',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_ocean_swflx_vis': ['ocean_model',
 'total_ocean_swflx_vis',
 'total_ocean_swflx_vis',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'total_volume_seawater': ['ocean_model',
 'total_volume_seawater',
 'total_volume_seawater',
 'ocean_scalar',
 'all',
 '.true.',
 'none',
 '2'],
'tx_trans': ['ocean_model',
 'tx_trans',
 'tx_trans',
 'ocean',
 'all',
 '.true.',
 'none',
 '2'],
'tx_trans_int_z': ['ocean_model',
 'tx_trans_int_z',
 'tx_trans_int_z',
 'ocean_month',
 'all',
 '.true.',
 'none',
 '2'],
'ty_trans': ['ocean_model',
 'ty_trans',
 'ty_trans',
 'ocean',
 'all',
 '.true.',
 'none',
 '2'],
'ty_trans_int_z': ['ocean_model',
 'ty_trans_int_z',
 'ty_trans_int_z',
 'ocean_month',
 'all',
 '.true.',
```

```
  'none',
  '2'],
 'ty_trans_rho': ['ocean_model',
  'ty_trans_rho',
  'ty_trans_rho',
  'ocean',
  'all',
  '.true.',
  'none',
  '2'],
 'u': ['ocean_model',
  'u',
  'u',
  'oceankerg_%4yr_%3dy',
  'all',
  '.true.',
  '-280,80,-81,-30,-1,-1',
  '2'],
 'v': ['ocean_model',
  'v',
  'v',
  'oceankerg_%4yr_%3dy',
  'all',
  '.true.',
  '-280,80,-81,-30,-1,-1',
  '2'],
 'wt': ['ocean_model',
  'wt',
  'wt',
  'oceankerg_%4yr_%3dy',
  'all',
  '.true.',
  '-280,80,-81,-30,-1,-1',
  '2']}
```

```
import tqdm
```

```
ls -1 /g/data3/hh5/tmp/cosima/mom01v5/KDS75_PI/output372
```

```
[0m[01;32mconfig.yaml[0m*
[01;32mdata_table[0m*
[01;32mdiag_table[0m*
[01;32mfield_table[0m*
ice_month.nc
[01;32minput.nml[0m*
logfile.000000.out
mom.err
mom.out
ocean__0096_184.nc
ocean__0096_189.nc
ocean__0096_194.nc
ocean__0096_199.nc
ocean__0096_204.nc
ocean__0096_209.nc
ocean__0096_214.nc
ocean__0096_219.nc
ocean__0096_224.nc
```

```
ocean__0096_229.nc
ocean__0096_234.nc
ocean__0096_239.nc
ocean__0096_244.nc
ocean__0096_249.nc
ocean__0096_254.nc
ocean__0096_259.nc
ocean__0096_264.nc
ocean__0096_269.nc
ocean_grid.nc
ocean_month.nc
ocean.nc
ocean_scalar.nc
time_stamp.out
[m
```

```
cd /g/data3/hh5/tmp/cosima/mom01v5/
```

```
/g/data3/hh5/tmp/cosima/mom01v5
```

```
ls
```

```
[0m[01;34mfigures[0m/  [01;34mGFDL50[0m/  [01;34mKDS75[0m/  [01;34mKDS75_PI[0m/  [01;
↪34mKDS75_wind[0m/  MOM01_Diagnostics.ipynb
[m
```

```
cd GFDL50/
```

```
/g/data3/hh5/tmp/cosima/mom01v5/GFDL50
```

```
ls
```

```
[0m[01;32mgfdl50.rho.384_403.ncra.nc[0m*     [01;34moutput346[0m/  [01;
↪34moutput367[0m/  [01;34moutput388[0m/
[01;32mgfdl50.uvwt.384_403.ncra.nc[0m*    [01;34moutput347[0m/  [01;34moutput368[0m/ ␣
↪[01;34moutput389[0m/
[01;32mgfdl50.wt_dia.384_403.ncra.nc[0m*  [01;34moutput348[0m/  [01;34moutput369[0m/ ␣
↪[01;34moutput390[0m/
[01;34moutput328[0m/                      [01;34moutput349[0m/  [01;34moutput370[0m/ ␣
↪[01;34moutput391[0m/
[01;34moutput329[0m/                      [01;34moutput350[0m/  [01;34moutput371[0m/ ␣
↪[01;34moutput392[0m/
[01;34moutput330[0m/                      [01;34moutput351[0m/  [01;34moutput372[0m/ ␣
↪[01;34moutput393[0m/
[01;34moutput331[0m/                      [01;34moutput352[0m/  [01;34moutput373[0m/ ␣
↪[01;34moutput394[0m/
[01;34moutput332[0m/                      [01;34moutput353[0m/  [01;34moutput374[0m/ ␣
↪[01;34moutput395[0m/
[01;34moutput333[0m/                      [01;34moutput354[0m/  [01;34moutput375[0m/ ␣
↪[01;34moutput396[0m/
[01;34moutput334[0m/                      [01;34moutput355[0m/  [01;34moutput376[0m/ ␣
↪[01;34moutput397[0m/
[01;34moutput335[0m/                      [01;34moutput356[0m/  [01;34moutput377[0m/ ␣
↪[01;34moutput398[0m/
[01;34moutput336[0m/                      [01;34moutput357[0m/  [01;34moutput378[0m/ ␣
↪[01;34moutput399[0m/
```

```
[01;34moutput337[0m/                          [01;34moutput358[0m/  [01;34moutput379[0m/ ␣
↪[01;34moutput400[0m/
[01;34moutput338[0m/                          [01;34moutput359[0m/  [01;34moutput380[0m/ ␣
↪[01;34moutput401[0m/
[01;34moutput339[0m/                          [01;34moutput360[0m/  [01;34moutput381[0m/ ␣
↪[01;34moutput402[0m/
[01;34moutput340[0m/                          [01;34moutput361[0m/  [01;34moutput382[0m/ ␣
↪[01;34moutput403[0m/
[01;34moutput341[0m/                          [01;34moutput362[0m/  [01;34moutput383[0m/ ␣
↪SimulationMetrics.npz
[01;34moutput342[0m/                          [01;34moutput363[0m/  [01;34moutput384[0m/ ␣
↪TauX_Last5Years.nc
[01;34moutput343[0m/                          [01;34moutput364[0m/  [01;34moutput385[0m/
[01;34moutput344[0m/                          [01;34moutput365[0m/  [01;34moutput386[0m/
[01;34moutput345[0m/                          [01;34moutput366[0m/  [01;34moutput387[0m/
[m
```

```
cd output328/
```

```
/g/data3/hh5/tmp/cosima/mom01v5/GFDL50/output328
```

```
ls
```

```
[0m[30;43mconfig.yaml[0m*  [30;43mice_month.nc[0m*        [30;43mmom.out[0m*          ␣
↪[30;43mtime_stamp.out[0m*
[30;43mdata_table[0m*   [30;43minput.nml[0m*           [30;43mocean_grid.nc[0m*
[30;43mdiag_table[0m*   [30;43mlogfile.000000.out[0m*  [30;43mocean_month.nc[0m*
[30;43mfield_table[0m*  [30;43mmom.err[0m*             [30;43mocean_scalar.nc[0m*
[m
```

```
cat time_stamp.out
```

```
56   1   1   0   0   0  Jan
56   4   1   0   0   0  Apr
```

APE-MOM paper figures

This is a compilation of other diagnostic files to hold final figures for the APE-MOM paper.

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
    import os
from netCDF4 import Dataset
from mpl_toolkits.basemap import Basemap, shiftgrid
```

```
# function to do latitude shifting:
def shifted(input,geolon):
    ny,nx = geolon.shape
    shifted = np.zeros(input.shape)
    for j in range(ny):
```

```
        ii = np.max(np.where(geolon[j,:]<-180)) + 1
        shifted[j,(1440-ii):] = input[j,:ii]
        shifted[j,:(1440-ii)] = input[j,ii:]
    return shifted
```

```
datadir = '/g/data1/v45/APE-MOM/gfdl_nyf_1080_'
```

Figure 2 - WindStress.pdf

```
# Put all in this figure
cols = ['k','g','m','c']
lines = ['-','--','-.','-']
exp_names = ['cp','UP','SH','PI']
labels = ['Control','Strong','Shift','Strong/Shift']
plt.figure(figsize(5,4))

ii=-1
for fn in exp_names:
    ii+=1
    filename = datadir+fn+'/ocean_tau_x_540-549.nc'
    print(filename)
    nc = Dataset(filename)
    tau = nc.variables['tau_x'][:,:,:]
    lat = nc.variables['yu_ocean'][:]
    nc.close()

    tau_bar = np.mean(tau,axis=2)[0,:]

    plt.plot(lat,tau_bar,linestyle=lines[ii],color=cols[ii],linewidth=1.5,
↪label=labels[ii])

plt.xlim([-75,-35])
plt.ylim([-0.02,0.2])
plt.xlabel('Latitude ($^\circ$N)')
plt.ylabel('Stress (N m$^{-2}$)')
plt.legend(loc=8,fontsize=10)

plt.show()
```

```
/g/data1/v45/APE-MOM/gfdl_nyf_1080_cp/ocean_tau_x_540-549.nc
/g/data1/v45/APE-MOM/gfdl_nyf_1080_UP/ocean_tau_x_540-549.nc
/g/data1/v45/APE-MOM/gfdl_nyf_1080_SH/ocean_tau_x_540-549.nc
/g/data1/v45/APE-MOM/gfdl_nyf_1080_PI/ocean_tau_x_540-549.nc
```

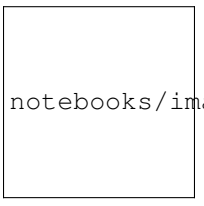


Figure 3 - MeanFields.pdf

```
# Put all in this figure
plt.figure(figsize(9.5,12))
x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)

datafile = Dataset(datadir+'cp/output501/ocean__501_363.nc','r')
geolon_c=datafile.variables['geolon_c'][:]
geolat_c=datafile.variables['geolat_c'][:]
geolon_t=datafile.variables['geolon_t'][:]
geolat_t=datafile.variables['geolat_t'][:]
u0=datafile.variables['u'][0,0,:,:]
v0=datafile.variables['v'][0,0,:,:]
datafile.close()

lon = shifted(geolon_c,geolon_c)
lat = shifted(geolat_c,geolon_c)

uv = (u0**2.0 + v0**2.0)**0.5
uv = shifted(uv,geolon_c)

plt.subplot(312)
lev = np.linspace(0,1.0,21)
map = Basemap(projection='mbtfpq',lon_0 = 0,resolution='l')
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
map.drawparallels(np.arange(-60.,61.,30.),labels=[True,False,False,False])
map.drawmeridians(np.arange(-180.,181.,90.),labels=[False,False,False,True])
X, Y = map(lon,lat)
map.contourf(X,Y,uv, 20, cmap=plt.cm.hot,levels=lev,extend='both') #CMRmap_r, gist_
↪stern_r
cb = plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('m s$^{-1}$')
plt.title('(b)')

datafile = Dataset(datadir+'cp/mean_ssh.nc','r')
sea_level=datafile.variables['sea_level'][0,:,:]
datafile.close()


datafile = Dataset(datadir+'cp/rho_ann_avge.nc','r')
pot_rho_2=datafile.variables['pot_rho_2'][0,0,:,:]
datafile.close()

lon = shifted(geolon_t,geolon_t)
lat = shifted(geolat_t,geolon_t)
ssh = shifted(sea_level,geolon_t)
rho = shifted(pot_rho_2,geolon_t)

plt.subplot(311)
lev = np.linspace(-2.0,1.0,11)
print (np.diff(lev))
map = Basemap(projection='mbtfpq',lon_0 = 0,resolution='l')
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
map.drawparallels(np.arange(-60.,61.,30.),labels=[True,False,False,False])
map.drawmeridians(np.arange(-180.,181.,90.),labels=[False,False,False,True])
X, Y = map(lon,lat)
```

```
map.contour(X,Y,ssh, colors='k',linewidths=0.5,levels=lev) #CMRmap_r, gist_stern_r
lev = np.linspace(1029.8,1037.5,21)
map.contourf(X,Y,rho, cmap=plt.cm.rainbow_r,levels=lev,extend='both') #CMRmap_r, gist_
↪stern_r
cb = plt.colorbar(orientation='vertical',shrink = 0.7,ticks=[1030, 1032, 1034, 1036])
cb.ax.set_xlabel('kg m$^{-3}$')
cb.ax.set_yticklabels(['1030', '1032', '1034', '1036'])
#cb.ax.yaxis.set_major_formatter(x_formatter)
plt.title('(a)')

nc = Dataset(datadir+'cp/resid_psi_rho.nc')
rho = nc.variables['potential_density'][:]
lat = nc.variables['latitude'][:]
psi = nc.variables['resid_psi'][:,:,:]
nc.close()

psi_bar_cp = np.mean(psi,0)
plt.subplot(313)
clev = np.arange(-20,20,2)
plt.contourf(lat,rho,psi_bar_cp,cmap=plt.cm.PiYG,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('Sv')
plt.contour(lat,rho,psi_bar_cp,levels=clev,colors='k',linewidths=0.25)
plt.contour(lat,rho,psi_bar_cp,levels=[0.0,],colors='k',linewidths=0.5)
plt.gca().invert_yaxis()
plt.gca().yaxis.set_major_formatter(x_formatter)
plt.ylim((1037.2,1033))
plt.ylabel('Potential Density (kg m$^{-3}$)')
plt.xlabel('Latitude ($^\circ$N)')
plt.xlim([-75,85])
plt.title('(c)')
plt.gca().text(50, 1034.5, 'Upper Cell',color='k',
        bbox={'facecolor':'white', 'alpha':0.95, 'pad':5})
plt.gca().annotate('', xy=(40, 1036), xytext=(49, 1034.6),
            arrowprops=dict(facecolor='black', shrink=0.05,width=2,headwidth=8),
            )
plt.gca().annotate('', xy=(-20, 1036), xytext=(51.4, 1034.6),
            arrowprops=dict(facecolor='black', shrink=0.05,width=2,headwidth=8),
            )
plt.gca().text(-70, 1035, 'Lower Cell',color='k',
        bbox={'facecolor':'white', 'alpha':0.95, 'pad':5})
plt.gca().annotate('', xy=(-65, 1036.95), xytext=(-69, 1035.05),
            arrowprops=dict(facecolor='black', shrink=0.05,width=2,headwidth=8),
            )
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/numpy/ma/core.
↪py:6468: RuntimeWarning: overflow encountered in power
  result = np.where(m, fa, umath.power(fa, fb)).view(basetype)
```

```
---------------------------------------------------------------------------

OSError                                   Traceback (most recent call last)

<ipython-input-21-70363a142e2b> in <module>()
     36
     37
---> 38 datafile = Dataset(datadir+'cp/rho_ann_avge.nc','r')
```

```
     39 pot_rho_2=datafile.variables['pot_rho_2'][0,0,:,:]
     40 datafile.close()


netCDF4/_netCDF4.pyx in netCDF4._netCDF4.Dataset.__init__ (netCDF4/_netCDF4.c:13992)()


OSError: No such file or directory
```

notebooks/images/APE-MOMFigures_1.png

Figure 4 - BasicStats.pdf

```
###### Look at temporal response
cols = ['k','g','m','c']
lines = ['-','--','-.','-']
exp_names = ['cp','UP','SH','PI']
labels = ['Control','Strong','Shift','Strong/Shift']
plt.figure(figsize(5,10))


# average between i1 and i2
i1 = 600
i2=610
j1 = 230
j2=235
ii=-1
for fn in exp_names:
    ii+=1
    filename = datadir+fn+'/resid_psi_rho.nc'
    nc = Dataset(filename)
    T = nc.variables['time'][:]
    rho = nc.variables['potential_density'][:]
    lat = nc.variables['latitude'][:]
    psi = nc.variables['resid_psi'][:,55:,i1:i2]
    psj = nc.variables['resid_psi'][:,:,j1:j2]
    nc.close()

    psi_bar = np.mean(psi,2)
    psi_ts = np.max(psi_bar,1)

    # Add year 500
    T = np.insert(T,0,500.)
    psi_ts = np.insert(psi_ts,0,16.6)
    plt.subplot(311)
    plt.plot(T,psi_ts,linestyle=lines[ii],color=cols[ii],linewidth=1.5)

    # do lower cell
    psj_bar = np.mean(psj,2)
    psj_ts = -np.min(psj_bar[:,50:],1)
    psj_ts = np.insert(psj_ts,0,6.5)
```

```
    plt.subplot(312)
    plt.plot(T,psj_ts,linestyle=lines[ii],color=cols[ii],label=labels[ii],linewidth=1.
↪5)

    filename = datadir+fn+'/DPTrans.npz'
    data=np.load(filename)
    Time = data['time']
    DPTrans = data['DPTrans']
    nyrs = Time.shape[0]//20
    DPTrans_AA = np.reshape(DPTrans,(nyrs,20)).mean(1)
    Time_AA = np.reshape(Time,(nyrs,20)).mean(1)
    if ii==0:
        DP500 = DPTrans_AA[14]
        T500 =Time_AA[14]
    else:
        DPTrans_AA = np.insert(DPTrans_AA,0,DP500)
        Time_AA = np.insert(Time_AA,0,T500)
    plt.subplot(313)
    plt.plot(Time_AA,DPTrans_AA,linestyle=lines[ii],color=cols[ii],linewidth=1.5)

    ## THESE PANELS DELETED AT LAST MOMENT!
    #plt.subplot(222)
    #x1 = np.mean(np.reshape(psi_ts[-50:],(10,5)),axis=1)
    #y1 = np.mean(np.reshape(DPTrans_AA[-50:],(10,5)),axis=1)
    #plt.plot(x1,y1,'.',linestyle='-',color=cols[ii],markersize=10,linewidth=0.75)


    #plt.subplot(224)
    #x1 = np.mean(np.reshape(psj_ts[-50:],(10,5)),axis=1)
    #plt.plot(x1,y1,'.',linestyle='-',color=cols[ii],label=labels[ii],markersize=10,
↪linewidth=0.75)

plt.legend()
print ("Bounding latitudes of psi average:", lat[i1], lat[i2])
print ("Bounding latitudes of psi_lower average:", lat[j1], lat[j2])

plt.subplot(311)
plt.ylabel('Upper Overturning (Sv)')
#plt.xlabel('Time (years)')
plt.title('(a)')

plt.subplot(313)
plt.ylabel('DP Transport (Sv)')
plt.xlabel('Time (years)')
plt.title('(c)')
plt.xlim([500,550])

plt.subplot(312)
plt.ylabel('Lower Overturning (Sv)')
#plt.xlabel('Time (years)')
plt.title('(b)')
plt.legend(fontsize=10)


#plt.savefig('figures/BasicStats.pdf')
```

```
--------------------------------------------------------------------------
```

```
IndexError                                Traceback (most recent call last)

<ipython-input-28-480916526589> in <module>()
     48     Time_AA = np.reshape(Time,(nyrs,20)).mean(1)
     49     if ii==0:
---> 50         DP500 = DPTrans_AA[14]
     51         T500 =Time_AA[14]
     52     else:


IndexError: index 14 is out of bounds for axis 0 with size 1
```

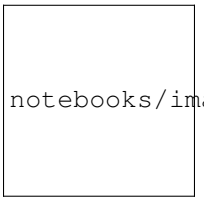


Figure 5 - MLD.pdf

```
exp_names = ['cp','UP','SH','PI']
labels = ['(a)','(b)','(c)','(d)']
plt.figure(figsize(10,10))

ii=-1
levels = np.linspace(0,2000,50)
for fn in exp_names:
    ii += 1
    filename = datadir+fn+'/ocean_mld_510-519.nc'
    nc = Dataset(filename)
    mld = nc.variables['mld'][0,:,:]
    #yt = nc.variables['yt_ocean'][:]
    #xt = nc.variables['xt_ocean'][:]
    nc.close()

    plt.subplot(2,2,ii+1)
    map = Basemap(projection='spstere',boundinglat=-45,lon_0=90,resolution='l',
↪round=True)
    map.drawcoastlines(linewidth=0.25)
    map.fillcontinents(color='gray',lake_color='gray')
    # draw parallels and meridians.
    map.drawparallels(np.arange(-80.,81.,15.))
    map.drawmeridians(np.arange(-160.,181.,40.),labels=[True,True,True,True])
    X, Y = map(geolon_t,geolat_t)
    p1=map.contourf(X,Y,mld, cmap=plt.cm.CMRmap_r, levels=levels, extend='both')
↪#CMRmap_r, gist_stern_r
    plt.title(labels[ii])


ax3 = plt.axes([0.92,0.38,0.015,0.25])
cb = plt.colorbar(p1,cax=ax3,orientation='vertical',ticks=[0, 500, 1000, 1500, 2000] )
cb.ax.set_xlabel('MLD (m)')

#plt.savefig('figures/MLD.pdf')
#plt.savefig('figures/MLD.png',dpi=220)
```

```
<matplotlib.text.Text at 0x7fb683cceb70>
```

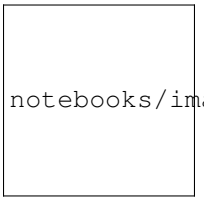


Figure 6 - GlobalEnergy.pdf

```
# Look at temporal response
cols = ['k','g','m','c']
lines = ['-','--','-.','-']
exp_names = ['cp','UP','SH','PI']
labels = ['Control','Strong','Shift','Strong/Shift']
plt.figure(figsize=(10,10))
fig, ((ax0, ax1), (ax2, ax3), (ax4, ax5)) = plt.subplots(3,2,figsize=(10,10))


ii=-1
for fn in exp_names:
    ii += 1
    filename = datadir+fn+'/ape_v1.nc'
    nc = Dataset(filename)
    T = nc.variables['T'][:]/365.
    APE = nc.variables['Available_Potential_Energy']
    KE = nc.variables['Kinetic_Energy']
    Phi_tau = nc.variables['Phi_tau'][:]
    Phi_z = nc.variables['Phi_z'][:]
    Phi_d1 = nc.variables['Phi_d1'][:]
    Phi_d2 = nc.variables['Phi_d2'][:]
    Phi_c1 = nc.variables['Phi_c1'][:]
    Phi_c2 = nc.variables['Phi_c2'][:]
    Phi_b1 = nc.variables['Phi_b1'][:]
    Phi_b2 = nc.variables['Phi_b2'][:]

    nyrs = T.shape[0]/73
    T_AA = T[36::73]
    APE_AA = np.reshape(APE,(nyrs,73)).mean(1)
    KE_AA = np.reshape(KE,(nyrs,73)).mean(1)
    Phi_tau_AA = np.reshape(Phi_tau,(nyrs,73)).mean(1)
    Phi_b_AA = np.reshape(Phi_b1-Phi_b2,(nyrs,73)).mean(1)
    Phi_z_AA = np.reshape(Phi_z,(nyrs,73)).mean(1)
    Phi_d_AA = np.reshape(Phi_d1-Phi_d2,(nyrs,73)).mean(1)
    Phi_c_AA = np.reshape(Phi_c1-Phi_c2,(nyrs,73)).mean(1)
    nc.close()
    dAPEdt = Phi_z_AA +Phi_d_AA + Phi_c_AA + Phi_b_AA


    plt.subplot(421)
    plt.plot(T_AA,KE_AA,linestyle=lines[ii],color=cols[ii],label=labels[ii],
↪linewidth=1.5)
    plt.subplot(422)
```

```
    plt.plot(T_AA,APE_AA,linestyle=lines[ii],color=cols[ii],label=labels[ii],
↪linewidth=1.5)

    plt.subplot(423)
    plt.plot(T_AA,Phi_tau_AA,linestyle=lines[ii],color=cols[ii],label=labels[ii],
↪linewidth=1.5)
    plt.subplot(424)
    plt.plot(T_AA,Phi_b_AA,linestyle=lines[ii],color=cols[ii],label=labels[ii],
↪linewidth=1.5)
    plt.subplot(425)
    plt.plot(T_AA,Phi_z_AA,linestyle=lines[ii],color=cols[ii],label=labels[ii],
↪linewidth=1.5)
    plt.subplot(426)
    plt.plot(T_AA,Phi_d_AA,linestyle=lines[ii],color=cols[ii],label=labels[ii],
↪linewidth=1.5)
    plt.subplot(427)
    plt.plot(T_AA,Phi_c_AA,linestyle=lines[ii],color=cols[ii],label=labels[ii],
↪linewidth=1.5)
    plt.subplot(428)
    plt.plot(T_AA,dAPEdt,linestyle=lines[ii],color=cols[ii],label=labels[ii],
↪linewidth=1.5)

plt.subplot(421)
plt.title('(a)')
plt.ylabel('KE (J/kg)')
plt.xlim([498,550])

plt.subplot(422)
plt.title('(b)')
plt.ylabel('APE (J/kg)')
plt.xlim([498,550])

plt.subplot(423)
plt.title('(c)')
plt.ylabel('Wind Power Input (W/kg)')
plt.xlim([498,550])

plt.subplot(424)
plt.title('(d)')
plt.ylabel('Buoyancy Power Input (W/kg)')
#ax3.set_xlabel('Time (years)')
plt.xlim([498,550])
plt.legend(fontsize=10)

plt.subplot(425)
plt.title('(e)')
plt.ylabel('KE to APE Conversion (W/kg)')
#plt.xlabel('Time (years)')
plt.xlim([498,550])

plt.subplot(426)
plt.title('(f)')
plt.ylabel('Diffusion (W/kg)')
plt.xlim([498,550])
#plt.tick_params(labelbottom='off')

plt.subplot(427)
plt.title('(g)')
```

```
plt.ylabel('Convection (W/kg)')
plt.xlabel('Time (years)')
plt.xlim([498,550])


plt.subplot(428)
plt.title('(h)')
plt.ylabel('dAPE/dt (W/kg)')
plt.xlabel('Time (years)')
plt.xlim([498,550])

#plt.savefig('figures/GlobalEnergy.pdf')
```

```
---------------------------------------------------------------------------

OSError                                   Traceback (most recent call last)

<ipython-input-31-6041b02bdcf2> in <module>()
     12     ii += 1
     13     filename = datadir+fn+'/ape_v1.nc'
---> 14     nc = Dataset(filename)
     15     T = nc.variables['T'][:]/365.
     16     APE = nc.variables['Available_Potential_Energy']


netCDF4/_netCDF4.pyx in netCDF4._netCDF4.Dataset.__init__ (netCDF4/_netCDF4.c:13992)()


OSError: No such file or directory
```

```
<matplotlib.figure.Figure at 0x7fb67d4d5780>
```

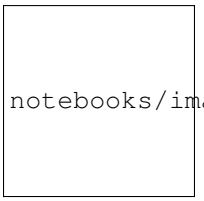


Figure 7 - GlobalLAPE.pdf

```
data=np.load(datadir+'cp/lape.npz' )
lape_cp_mean = data['lape_cp_mean']

lon = shifted(geolon_t,geolon_t)
lat = shifted(geolat_t,geolon_t)
lape_cp_mean = shifted(lape_cp_mean,geolon_t)

fig=plt.figure(figsize(13.5,6))
levels = np.linspace(np.log10(10/1030.),np.log10(30000/1030.),66)
map = Basemap(projection='mbtfpq',lon_0 = 0,resolution='l')
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
map.drawparallels(np.arange(-80.,81.,20.),labels=[True,False,False,False])
map.drawmeridians(np.arange(-180.,181.,60.),labels=[False,False,False,True])
```

```
X, Y = map(lon,lat)
map.contourf(X,Y,np.log10(lape_cp_mean/1030), cmap=plt.cm.CMRmap_r, levels=levels,␣
↪extend='both') #CMRmap_r, gist_stern_r
cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-2, -1, 0, 1])
cb.ax.set_xlabel('log$_{10}$(APE) (J/kg)')

#plt.savefig('figures/GlobalLAPE.pdf')
#plt.savefig('figures/GlobalLAPE.png',dpi=144)
```

```
---------------------------------------------------------------------------

FileNotFoundError                         Traceback (most recent call last)

<ipython-input-32-ae6bb06de52e> in <module>()
----> 1 data=np.load(datadir+'cp/lape.npz' )
      2 lape_cp_mean = data['lape_cp_mean']
      3
      4 lon = shifted(geolon_t,geolon_t)
      5 lat = shifted(geolat_t,geolon_t)


/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/numpy/lib/npyio.py in␣
↪load(file, mmap_mode, allow_pickle, fix_imports, encoding)
    368     own_fid = False
    369     if isinstance(file, basestring):
--> 370         fid = open(file, "rb")
    371         own_fid = True
    372     elif is_pathlib_path(file):


FileNotFoundError: [Errno 2] No such file or directory: '/g/data1/v45/APE-MOM/gfdl_
↪nyf_1080_cp/lape.npz'
```

Figure 8 - SH_LAPE.pdf

```
cols = ['k','g','m','c']
lines = ['-','--','-.','-']
exp_names = ['cp','UP','SH','PI']
labels = ['Control','Strong','Shift','Strong/Shift']
plt.figure(figsize(10,10))

plt.subplot(221)
data=np.load(datadir+'UP/lape.npz' )
lape_UP_diff = data['lape_UP_diff']/1030.    ## Divide by rho_0
levels = np.linspace(-5,5,66)

map = Basemap(projection='spstere',boundinglat=-45,lon_0=90,resolution='l',round=True)
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
# draw parallels and meridians.
map.drawparallels(np.arange(-80.,81.,15.))
map.drawmeridians(np.arange(-160.,181.,40.),labels=[True,True,True,True])
X, Y = map(geolon_t,geolat_t)
map.contourf(X,Y,lape_UP_diff, cmap=plt.cm.PuOr, levels=levels, extend='both')
↪#CMRmap_r, gist_stern_r
#cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-4000, -2000, 0, 2000,␣
↪4000 ])
```

```
plt.title('(a)')


plt.subplot(222)
data=np.load(datadir+'SH/lape.npz' )
lape_SH_diff = data['lape_SH_diff']/1030.    ## Divide by rho_0

map = Basemap(projection='spstere',boundinglat=-45,lon_0=90,resolution='l',round=True)
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
# draw parallels and meridians.
map.drawparallels(np.arange(-80.,81.,15.))
map.drawmeridians(np.arange(-160.,181.,40.),labels=[True,True,True,True])
X, Y = map(geolon_t,geolat_t)
map.contourf(X,Y,lape_SH_diff, cmap=plt.cm.PuOr, levels=levels, extend='both')
 ↪#CMRmap_r, gist_stern_r
#cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-4000, -2000, 0, 2000,␣
 ↪4000 ])
plt.title('(b)')


plt.subplot(223)
data=np.load(datadir+'PI/lape.npz' )
lape_PI_diff = data['lape_PI_diff']/1030.    ## Divide by rho_0

map = Basemap(projection='spstere',boundinglat=-45,lon_0=90,resolution='l',round=True)
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
# draw parallels and meridians.
map.drawparallels(np.arange(-80.,81.,15.))
map.drawmeridians(np.arange(-160.,181.,40.),labels=[True,True,True,True])
X, Y = map(geolon_t,geolat_t)
p1=map.contourf(X,Y,lape_PI_diff, cmap=plt.cm.PuOr, levels=levels, extend='both')
 ↪#CMRmap_r, gist_stern_r
#cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-4000, -2000, 0, 2000,␣
 ↪4000 ])
plt.title('(c)')

lape_zm_UP = np.mean(lape_UP_diff,1)
lape_zm_PI = np.mean(lape_PI_diff,1)
lape_zm_SH = np.mean(lape_SH_diff,1)

plt.subplot(224)
plt.plot(geolat_t[:498,0],lape_zm_UP[:498],linestyle=lines[1],color=cols[1],
 ↪label=labels[1],linewidth=1.5)
plt.plot(geolat_t[:498,0],lape_zm_SH[:498],linestyle=lines[2],color=cols[2],
 ↪label=labels[2],linewidth=1.5)
plt.plot(geolat_t[:498,0],lape_zm_PI[:498],linestyle=lines[3],color=cols[3],
 ↪label=labels[3],linewidth=1.5)
plt.legend(fontsize=10)
plt.xlim([-76,-45])
plt.ylim([-2.5, 2.5])
plt.xlabel('Latitude ($^\circ$N)')
plt.ylabel('$\Delta$ APE (J/kg)')
plt.title('(d)')

ax3 = plt.axes([0.09,0.39,0.015,0.23])
cb = plt.colorbar(p1,cax=ax3,orientation='vertical',ticks=[-5, -2.5, 0, 2.5, 5])
```

```
cb.ax.set_ylabel('$\Delta$ APE (J/kg)')

#plt.savefig('figures/SH_LAPE.pdf')
#plt.savefig('figures/SH_LAPE.png',dpi=220)
```

```
---------------------------------------------------------------------------

FileNotFoundError                         Traceback (most recent call last)

<ipython-input-42-9d30e430207b> in <module>()
      6
      7 plt.subplot(221)
----> 8 data=np.load(datadir+'UP/lape.npz' )
      9 lape_UP_diff = data['lape_UP_diff']/1030.    ## Divide by rho_0
     10 levels = np.linspace(-5,5,66)


/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/numpy/lib/npyio.py in␣
↪load(file, mmap_mode, allow_pickle, fix_imports, encoding)
    368     own_fid = False
    369     if isinstance(file, basestring):
--> 370         fid = open(file, "rb")
    371         own_fid = True
    372     elif is_pathlib_path(file):


FileNotFoundError: [Errno 2] No such file or directory: '/g/data1/v45/APE-MOM/gfdl_
↪nyf_1080_UP/lape.npz'
```

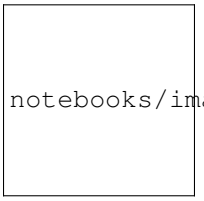


Figure 9 - NH_LAPE.pdf

```
cols = ['k','g','m','c']
lines = ['-','--','-.','-']
exp_names = ['cp','UP','SH','PI']
labels = ['Control','Strong','Shift','Strong/Shift']
plt.figure(figsize(10,3.5))

# bounds of meridional average
## 700 - 825 isolates subpolar gyre
l1 = 800
l2 = 1050
m1 = 920
m2 = 1340

plt.subplot(121)
data=np.load(datadir+'UP/lape_NH.npz' )
lape_UP_diff = data['lape_UP_diff']/1030.    ## Divide by rho_0
lape_UP_GIN = data['lape_UP_GIN']/1030.    ## Divide by rho_0
```

```
time = data['time']

lape_plot = np.mean(lape_UP_diff[39:,l1:l2,m1:m2],axis=0)

levels = np.linspace(-5,5,66)
map = Basemap(llcrnrlon=-33,llcrnrlat=63,urcrnrlon=39,urcrnrlat=72,resolution='i',
↪projection='cass',lon_0=-25,lat_0=66)
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
# draw parallels and meridians.
map.drawparallels(np.arange(-80.,81.,10.),labels=[True,False,False,True])
map.drawmeridians(np.arange(-180.,181.,20.),labels=[True,False,False,True])
X, Y = map(geolon_t[l1:l2,m1:m2],geolat_t[l1:l2,m1:m2])
map.contourf(X,Y,lape_plot, cmap=plt.cm.PuOr, levels=levels, extend='both') #CMRmap_r,
↪ gist_stern_r
cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-4, -2, 0, 2, 4 ])
cb.ax.set_xlabel('$\Delta$ APE\n (J/kg)')
plt.title('(a)')

plt.subplot(122)
data=np.load('data/SH/lape_NH.npz' )
lape_SH_GIN = data['lape_SH_GIN']/1030.    ## Divide by rho_0
data=np.load('data/PI/lape_NH.npz' )
lape_PI_GIN = data['lape_PI_GIN']/1030.    ## Divide by rho_0

plt.plot(time/365,lape_UP_GIN,color=cols[1],linestyle=lines[1],label=labels[1],
↪linewidth=1.5)
plt.plot(time/365,lape_SH_GIN,color=cols[2],linestyle=lines[2],label=labels[2],
↪linewidth=1.5)
plt.plot(time/365,lape_PI_GIN,color=cols[3],linestyle=lines[3],label=labels[3],
↪linewidth=1.5)

plt.legend(loc=2,fontsize=10)
plt.ylim([-0.5, 1])
plt.xlabel('Time (years)')
plt.ylabel('$\Delta$ APE (J/kg)')
plt.title('(b)')

#plt.savefig('figures/NH_LAPE.pdf')
#plt.savefig('figures/NH_LAPE.png',dpi=144)
```

```
---------------------------------------------------------------------------

FileNotFoundError                         Traceback (most recent call last)

<ipython-input-39-66dcf6f84cde> in <module>()
     13
     14 plt.subplot(121)
---> 15 data=np.load(datadir+'UP/lape_NH.npz' )
     16 lape_UP_diff = data['lape_UP_diff']/1030.    ## Divide by rho_0
     17 lape_UP_GIN = data['lape_UP_GIN']/1030.    ## Divide by rho_0


/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/numpy/lib/npyio.py in␣
↪load(file, mmap_mode, allow_pickle, fix_imports, encoding)
    368     own_fid = False
    369     if isinstance(file, basestring):
--> 370         fid = open(file, "rb")
```

```
    371         own_fid = True
    372     elif is_pathlib_path(file):


FileNotFoundError: [Errno 2] No such file or directory: '/g/data1/v45/APE-MOM/gfdl_
↪nyf_1080_UP/lape_NH.npz'
```

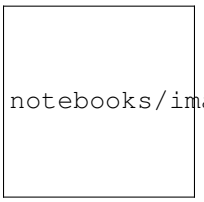


Figure 10 - ResidPsi.pdf

```
fig = plt.figure(figsize=(13,8))
exp_names = ['UP','SH','PI']
titles=['(a)','(b)','(c)','(d)','(e)','(f)']
lev = np.linspace(-20,20,10)
clev = np.linspace(-10,10,20)
x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)

# first do years 15-25
i1 = 15
i2 = 25
ii = 0
for fn in exp_names:
    ii += 1
    filename = 'data/'+fn+'/resid_psi_rho.nc'
    nc = Dataset(filename)
    T = nc.variables['time'][:]
    rho = nc.variables['potential_density'][:]
    lat = nc.variables['latitude'][:]
    psi = nc.variables['resid_psi'][:,:,:]
    nc.close()

    psi_bar = np.mean(psi[i1:i2,:,:],0) - psi_bar_cp
    plt.subplot(2,3,ii)
    plt.contourf(lat,rho,psi_bar,cmap=plt.cm.PRGn,levels=clev,extend='both')
    plt.contour(lat,rho,psi_bar_cp,levels=lev,colors='k',linewidths=0.25)
    plt.contour(lat,rho,psi_bar_cp,levels=[0, ],colors='k',linewidths=1)
    plt.gca().invert_yaxis()
    plt.ylim((1037.2,1033))
    plt.xlim([-75,85])
    plt.title(titles[ii-1])
    plt.gca().yaxis.set_major_formatter(x_formatter)

plt.subplot(231)
plt.ylabel('Potential Density (kg m$^{-3}$)')
# first do years 15-25
i1 = 40
i2 = 50
for fn in exp_names:
    ii += 1
    filename = datadir+fn+'/resid_psi_rho.nc'
```

```
    nc = Dataset(filename)
    T = nc.variables['time'][:]
    rho = nc.variables['potential_density'][:]
    lat = nc.variables['latitude'][:]
    psi = nc.variables['resid_psi'][:,:,:]
    nc.close()

    psi_bar = np.mean(psi[i1:i2,:,:],0) - psi_bar_cp
    plt.subplot(2,3,ii)
    p2=plt.contourf(lat,rho,psi_bar,cmap=plt.cm.PRGn,levels=clev,extend='both')
    plt.contour(lat,rho,psi_bar_cp,levels=lev,colors='k',linewidths=0.25)
    plt.contour(lat,rho,psi_bar_cp,levels=[0, ],colors='k',linewidths=1)
    plt.gca().invert_yaxis()
    plt.ylim((1037.2,1033))
    plt.xlabel('Latitude ($^\circ$N)')
    plt.xlim([-75,85])
    plt.title(titles[ii-1])
    plt.gca().yaxis.set_major_formatter(x_formatter)



plt.subplot(234)
plt.ylabel('Potential Density (kg m$^{-3}$)')

ax4 = plt.axes([0.935,0.3,0.011,0.4])
cb = plt.colorbar(p2,cax=ax4,orientation='vertical',ticks=[-10,-5,0,5,10])
cb.ax.set_xlabel('Sv')

#plt.savefig('figures/ResidPsi.pdf')
#plt.savefig('figures/ResidPsi.png',dpi=220)
```

```
---------------------------------------------------------------------------

OSError                                   Traceback (most recent call last)

<ipython-input-38-66a6c2c83696> in <module>()
     13     ii += 1
     14     filename = 'data/'+fn+'/resid_psi_rho.nc'
---> 15     nc = Dataset(filename)
     16     T = nc.variables['time'][:]
     17     rho = nc.variables['potential_density'][:]


netCDF4/_netCDF4.pyx in netCDF4._netCDF4.Dataset.__init__ (netCDF4/_netCDF4.c:13992)()


OSError: No such file or directory
```

```
<matplotlib.figure.Figure at 0x7fb685fb9358>
```

Figure 11 - DepthDensityPsi.pdf

```
## First, take CP and plot total, mean and eddy streamfunction in N and S hemispheres
fig, ax = plt.subplots(2,2,figsize=(6,5))
subplots_adjust(wspace=0)
```

```
clev = np.linspace(-20,20,41)
dlev = np.linspace(-10,10,30)
x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)
titles=['(a)','(b)','(c)','(d)']


filename = 'data/cp/rhoz_moc.nc'
nc = Dataset(filename)
T = nc.variables['time'][:]
rho = nc.variables['rhobin'][:]
z = nc.variables['st_ocean'][:]
trans_south = nc.variables['rho_z_trans_south'][:,:,:]
trans_north = nc.variables['rho_z_trans_north'][:,:,:]
nc.close()

# Load equatorial density file
filename = datadir+'cp/ocean_541-550.nc'
nc = Dataset(filename)
Equator_Rho = nc.variables['pot_rho_2'][0,:,497,:]
nc.close()
Eq_Max_Rho = np.ma.max(Equator_Rho,axis=1)

trans_south_total = np.mean(trans_south,0)
trans_north_total = np.mean(trans_north,0)

psi_south_total_cp = np.cumsum(trans_south_total[:,::-1],1)[:,::-1]
psi_north_total_cp = np.cumsum(trans_north_total[:,::-1],1)[:,::-1]

# Plot density contours in x-z space.
ax0=ax[1,0]
ax1=ax[1,1]
ax1.set_yticklabels('')
p1=ax0.contourf(rho,z,psi_south_total_cp,cmap=plt.cm.PiYG,levels=clev,extend='both')
ax0.contour(rho,z,psi_south_total_cp,colors='k',levels=clev[::3],linewidths=0.25)
ax0.contour(rho,z,psi_south_total_cp,colors='k',levels=[0,],linewidths=1)
ax0.plot(Eq_Max_Rho,z,'gray',linewidth=2)
ax0.invert_yaxis()
ax0.set_xlim((1037.2,1034.5))
ax0.set_ylim((4500,0))
ax0.set_yticks([4000,3000,2000, 1000, 0])
ax0.set_ylabel('Depth (m)')
ax0.set_xticks([1035,1036,1037])
ax0.xaxis.set_major_formatter(x_formatter)
mc=ax1.contourf(rho,z,-psi_north_total_cp,cmap=plt.cm.PiYG,levels=clev,extend='both')
ax1.contour(rho,z,-psi_north_total_cp,colors='k',levels=clev[::3],linewidths=0.25)
ax1.contour(rho,z,-psi_north_total_cp,colors='k',levels=[0,],linewidths=1)
ax1.plot(Eq_Max_Rho,z,'gray',linewidth=2)
ax1.invert_yaxis()
ax1.set_ylim((4500,0))
ax1.set_yticks([4000,3000,2000, 1000, 0])
ax1.set_xlim((1037.2,1034.5))
ax1.invert_xaxis()
ax1.set_xticks([1035,1036,1037])
ax1.xaxis.set_major_formatter(x_formatter)
ax1.set_title(titles[1],loc='left', horizontalalignment='center')
ax1.set_xlabel('Potential Density (kg m$^{-3}$)')#,loc='left', horizontalalignment=
↪'center')
ax0.set_xlabel('Potential Density (kg m$^{-3}$)')
```

```
# Plot density contours in x-z space.
fig.delaxes(ax[0,0])
fig.delaxes(ax[0,1])
ax2 = plt.axes([0.25,0.56,0.5, 0.36])
p1=ax2.contourf(rho,z,psi_south_total_cp + psi_north_total_cp,cmap=plt.cm.PiYG,
↪levels=clev,extend='both')
ax2.contour(rho,z,psi_south_total_cp + psi_north_total_cp,colors='k',levels=clev[::3],
↪linewidths=0.25)
#ax2.contour(rho,z,psi_south_total_cp + psi_north_total_cp,colors='k',levels=[0,],
↪linewidths=1)
ax2.invert_yaxis()
ax2.set_xlim((1037.2,1034.5))
ax2.set_ylim((4500,0))
ax2.set_yticks([4000,3000,2000, 1000, 0])
ax2.set_ylabel('Depth (m)')
ax2.set_xticks([1035,1036,1037])
ax2.xaxis.set_major_formatter(x_formatter)
ax2.set_title(titles[0])

ax3 = plt.axes([0.8,0.6,0.015,0.3])
cb = plt.colorbar(p1,cax=ax3,orientation='vertical',ticks=[-20,-10,0,10,20])
cb.ax.set_xlabel('Sv')

#plt.savefig('figures/DepthDensityPsi.pdf')
#plt.savefig('figures/DepthDensityPsi.png',dpi=220)
```

```
---------------------------------------------------------------------------

OSError                                   Traceback (most recent call last)

<ipython-input-36-5ef937ab748c> in <module>()
      9
     10 filename = 'data/cp/rhoz_moc.nc'
---> 11 nc = Dataset(filename)
     12 T = nc.variables['time'][:]
     13 rho = nc.variables['rhobin'][:]


netCDF4/_netCDF4.pyx in netCDF4._netCDF4.Dataset.__init__ (netCDF4/_netCDF4.c:13992)()


OSError: No such file or directory
```

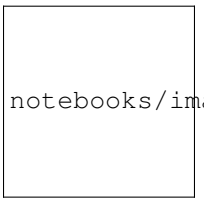


Figure 12 - DepthDensity2.pdf

```
## First, take CP and plot total, mean and eddy streamfunction in N and S hemispheres
fig, ax = plt.subplots(3,2,figsize=(6,8))
```

```
subplots_adjust(wspace=0)
clev = np.linspace(-20,20,41)
dlev = np.linspace(-10,10,30)
x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)
titles=['(a)','(b)','(c)','(d)']
# now loop over these
exp_names = ['UP','SH','PI']
ii=0
i1 = 40
i2 = 50
for fn in exp_names:
    ii += 1
    filename = datadir+fn+'/rhoz_moc.nc'
    nc = Dataset(filename)
    T = nc.variables['time'][:]
    rho = nc.variables['rhobin'][:]
    z = nc.variables['st_ocean'][:]
    trans_south = nc.variables['rho_z_trans_south'][:,:,:]
    trans_north = nc.variables['rho_z_trans_north'][:,:,:]
    nc.close()

    trans_south_total = np.mean(trans_south[i1:i2,:,:],0)
    trans_north_total = np.mean(trans_north[i1:i2,:,:],0)

    psi_south_total = np.cumsum(trans_south_total[:,::-1],1)[:,::-1]
    psi_north_total = np.cumsum(trans_north_total[:,::-1],1)[:,::-1]


    ax0=ax[ii-1,0]
    ax1=ax[ii-1,1]
    p2=ax0.contourf(rho,z,psi_south_total - psi_south_total_cp,cmap=plt.cm.PRGn,
↪levels=dlev,extend='both')
    ax0.contour(rho,z,psi_south_total_cp,colors='k',levels=clev[::3],linewidths=0.25)
    ax0.contour(rho,z,psi_south_total_cp,colors='k',levels=[0,],linewidths=1)
    ax0.plot(Eq_Max_Rho,z,'gray',linewidth=2)
    ax0.invert_yaxis()
    ax0.set_xlim((1037.2,1034.5))
    ax0.set_ylim((4500,0))
    ax0.set_xticks([1035,1036,1037])
    ax0.set_yticks([4000,3000,2000, 1000, 0])
    ax0.xaxis.set_major_formatter(x_formatter)
    ax0.set_ylabel('Depth (m)')
    mc=ax1.contourf(rho,z,-psi_north_total+psi_north_total_cp,cmap=plt.cm.PRGn,
↪levels=dlev,extend='both')
    ax1.contour(rho,z,-psi_north_total_cp,colors='k',levels=clev[::3],linewidths=0.25)
    ax1.contour(rho,z,-psi_north_total_cp,colors='k',levels=[0,],linewidths=1)
    ax1.plot(Eq_Max_Rho,z,'gray',linewidth=1)
    ax1.invert_yaxis()
    ax1.set_xlim((1037.2,1034.5))
    ax1.invert_xaxis()
    ax1.set_yticklabels('')
    ax1.set_ylim((4500,0))
    ax1.set_yticks([4000,3000,2000, 1000, 0])
    ax1.set_xticks([1035,1036,1037])
    ax1.xaxis.set_major_formatter(x_formatter)
    ax1.set_title(titles[ii-1],loc='left', horizontalalignment='center')

ax1.set_xlabel('Potential Density (kg m$^{-3}$)')#,loc='left', horizontalalignment=
↪'center')
```

```
ax0.set_xlabel('Potential Density (kg m$^{-3}$)')

ax4 = plt.axes([0.93,0.325,0.015,0.35])
cb = plt.colorbar(p2,cax=ax4,orientation='vertical',ticks=[-10,-5,0,5,10])
cb.ax.set_xlabel('Sv')

#plt.savefig('figures/DepthDensityPsi2.pdf')
#plt.savefig('figures/DepthDensityPsi2.png',dpi=220)
```

```
---------------------------------------------------------------------------

NameError                                 Traceback (most recent call last)

<ipython-input-35-abaf700d20b8> in <module>()
     31     ax0=ax[ii-1,0]
     32     ax1=ax[ii-1,1]
---> 33     p2=ax0.contourf(rho,z,psi_south_total - psi_south_total_cp,cmap=plt.cm.
↪PRGn,levels=dlev,extend='both')
     34     ax0.contour(rho,z,psi_south_total_cp,colors='k',levels=clev[::3],
↪linewidths=0.25)
     35     ax0.contour(rho,z,psi_south_total_cp,colors='k',levels=[0,],linewidths=1)


NameError: name 'psi_south_total_cp' is not defined
```

notebooks/images/APE-MOMFigures_8.png

Figure 13 - SH-EddyMean.pdf

```
## First, take CP and plot total, mean and eddy streamfunction in N and S hemispheres
clev = np.linspace(-30,30,41)
print (np.diff(clev[::3]))
dlev = np.linspace(-15,15,30)
x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)
titles=['(a)','(b)','(c)','(d)','(e)','(f)','(g)','(h)','(i)']

# Look at middle 10 years,
i1 = 20
i2 = 30
ii = 0
jj = 2 # 3rd decade
exp_names = ['UP','SH','PI']

# Get CP data
filename = datadir+'cp/rhoz_moc_mean.nc'
nc = Dataset(filename)
T = nc.variables['time'][:]
rho = nc.variables['rhobin'][:]
z = nc.variables['st_ocean'][:]
trans_south = nc.variables['rho_z_trans_south'][:,:,:]
```

```
nc.close()

trans_south_mean = np.mean(trans_south,0)
psi_south_mean_cp = np.cumsum(trans_south_mean[:,::-1],1)[:,::-1]
psi_south_eddy_cp = psi_south_total_cp - psi_south_mean_cp


fig, ax = plt.subplots(3,3,figsize=(10,10))

for fn in exp_names:
    ii += 1
    filename = 'data/'+fn+'/rhoz_moc.nc'
    nc = Dataset(filename)
    T = nc.variables['time'][:]
    rho = nc.variables['rhobin'][:]
    z = nc.variables['st_ocean'][:]
    trans_south = nc.variables['rho_z_trans_south'][:,:,:]
    nc.close()

    trans_south_total = np.mean(trans_south[i1:i2,:,:],0)

    psi_south_total = np.cumsum(trans_south_total[:,::-1],1)[:,::-1]

    filename = 'data/'+fn+'/rhoz_moc_mean.nc'
    nc = Dataset(filename)
    T = nc.variables['time'][:]
    rho = nc.variables['rhobin'][:]
    z = nc.variables['st_ocean'][:]
    trans_south = nc.variables['rho_z_trans_south'][:,:,:]
    nc.close()

    trans_south_mean = trans_south[jj,:,:]

    psi_south_mean = np.cumsum(trans_south_mean[:,::-1],1)[:,::-1]

    psi_south_eddy = psi_south_total - psi_south_mean

    ax0=ax[ii-1,0]
    ax1=ax[ii-1,1]
    ax2=ax[ii-1,2]
    p2=ax0.contourf(rho,z,psi_south_total - psi_south_total_cp,cmap=plt.cm.PRGn,
↪levels=dlev,extend='both')
    ax0.contour(rho,z,psi_south_total_cp,colors='k',levels=clev[::3],linewidths=0.25)
    ax0.contour(rho,z,psi_south_total_cp,colors='k',levels=[0,],linewidths=1)
    ax0.plot(Eq_Max_Rho,z,'gray',linewidth=2)
    ax0.invert_yaxis()
    ax0.set_xlim((1037.2,1034.5))
    ax0.set_ylim((4500,0))
    ax0.set_xticks([1035,1036,1037])
    ax0.set_yticks([4000,3000,2000, 1000, 0])
    ax0.xaxis.set_major_formatter(x_formatter)
    ax0.set_ylabel('Depth (m)')
    ax0.set_title(titles[(ii-1)*3])
    mc=ax1.contourf(rho,z,psi_south_mean - psi_south_mean_cp,cmap=plt.cm.PRGn,
↪levels=dlev,extend='both')
    ax1.contour(rho,z,psi_south_mean_cp,colors='k',levels=clev[::3],linewidths=0.25)
    ax1.contour(rho,z,psi_south_mean_cp,colors='k',levels=[0,],linewidths=1)
    ax1.plot(Eq_Max_Rho,z,'gray',linewidth=2)
```

```
    ax1.invert_yaxis()
    ax1.set_xlim((1037.2,1034.5))
    ax1.set_ylim((4500,0))
    ax1.set_xticks([1035,1036,1037])
    ax1.set_yticks([4000,3000,2000, 1000, 0])
    ax1.xaxis.set_major_formatter(x_formatter)
    ax1.set_title(titles[(ii-1)*3+1])
    mc=ax2.contourf(rho,z,psi_south_eddy- psi_south_eddy_cp,cmap=plt.cm.PRGn,
↪levels=dlev,extend='both')
    ax2.contour(rho,z,psi_south_eddy_cp,colors='k',levels=clev[::3],linewidths=0.25)
    #ax2.contour(rho,z,psi_south_eddy_cp,colors='k',levels=[0,],linewidths=1)
    ax2.plot(Eq_Max_Rho,z,'gray',linewidth=2)
    ax2.invert_yaxis()
    ax2.set_xlim((1037.2,1034.5))
    ax2.set_ylim((4500,0))
    ax2.set_xticks([1035,1036,1037])
    ax2.set_yticks([4000,3000,2000, 1000, 0])
    ax2.xaxis.set_major_formatter(x_formatter)
    ax2.set_title(titles[(ii-1)*3+2])

ax0.set_xlabel('Potential Density (kg m$^{-3}$)')
ax1.set_xlabel('Potential Density (kg m$^{-3}$)')
ax2.set_xlabel('Potential Density (kg m$^{-3}$)')

ax4 = plt.axes([0.92,0.35,0.015,0.4])
cb = plt.colorbar(p2,cax=ax4,orientation='vertical',ticks=[-15, -10,-5,0,5,10, 15])
cb.ax.set_xlabel('Sv')

#plt.savefig('figures/SH-EddyMean.pdf')
#plt.savefig('figures/SH-EddyMean.png',dpi=220)
```

```
[ 4.5  4.5  4.5  4.5  4.5  4.5  4.5  4.5  4.5  4.5  4.5  4.5  4.5]
```

```
---------------------------------------------------------------------------

NameError                                 Traceback (most recent call last)

<ipython-input-34-c9aad49be363> in <module>()
     24 trans_south_mean = np.mean(trans_south,0)
     25 psi_south_mean_cp = np.cumsum(trans_south_mean[:,::-1],1)[:,::-1]
---> 26 psi_south_eddy_cp = psi_south_total_cp - psi_south_mean_cp
     27
     28


NameError: name 'psi_south_total_cp' is not defined
```

MOM01 Salt Evaluations

plots surface and zonal average salts across simulations

computes anomalies relative to longterm average of WOA data

takes forever to run thru all expts, I suggest trying a subset

expts = ['GFDL50','KDS75']

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
# Load modules
    from netCDF4 import Dataset # to work with NetCDF files
import numpy as np
import matplotlib.pyplot as plt # to generate plots
from mpl_toolkits.basemap import Basemap # plot on map projections
from os.path import expanduser
home = expanduser("~") # Get users home directory
import os # operating system interface

import xarray
from xarray.ufuncs import square, sqrt
import scipy.interpolate

#import scipy
#sfrom scipy.interpolate import RegularGridInterpolator, griddata

#import dask
#from dask import delayed
#import dask.array as da
#import dask.dataframe as dd
#from dask.multiprocessing import get
#from dask.async import get_sync
#from dask.diagnostics import ProgressBar

#import pandas as pd
#from glob import glob
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
#pbar = ProgressBar()
#pbar.register()
```

```
# set up dask to initiate 4 parallel processors.
#this doesn't work yet
#dask.set_options(get=dask.multiprocessing.get, num_workers=4)
```

```
#extract the MOM model grid info - 50 and 75 vertical levels
hgrid_file ='/g/data1/v45/pas561/mom/archive/VertOverturn/mom01_unmasked_ocean_grid.nc
↪'
vgrid75_file ='/g/data1/v45/pas561/mom/archive/VertOverturn/kds75.uvwt.230-257.ncra.nc
↪'
vgrid50_file ='/g/data3/hh5/tmp/cosima/mom01v5/GFDL50/gfdl50.uvwt.384_403.ncra.nc'

# Extract the variables
nc = Dataset(hgrid_file, mode='r') # file handle, open in read only mode
geolon_t = nc.variables['geolon_t'][:]
geolat_t = nc.variables['geolat_t'][:]
area_t = nc.variables['area_t'][:]
dxt = nc.variables['dxt'][:]
dyt = nc.variables['dyt'][:]
nc.close() # close the file
print geolon_t.shape

nc = Dataset(vgrid75_file, mode='r') # file handle, open in read only mode
sw75_ocean = nc.variables['sw_ocean'][:]
st75_ocean = nc.variables['st_ocean'][:]
yt_ocean = nc.variables['yt_ocean'][:]
xt_ocean = nc.variables['xt_ocean'][:]
nc.close() # close the file

nc = Dataset(vgrid50_file, mode='r') # file handle, open in read only mode
sw50_ocean = nc.variables['sw_ocean'][:]
st50_ocean = nc.variables['st_ocean'][:]
yt_ocean = nc.variables['yt_ocean'][:]
nc.close() # close the file

print st75_ocean.shape
print st50_ocean.shape
```

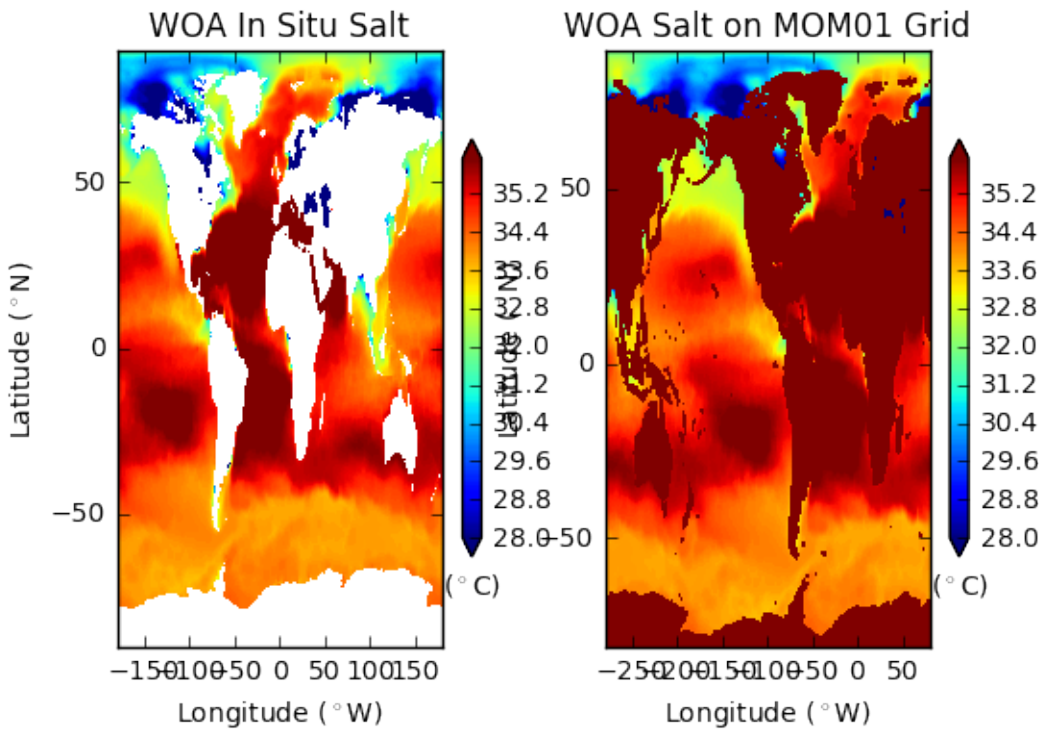
```
(2700, 3600)
(75,)
(50,)
```

```
#Get WOA observed Salt (In Situ) data and interpolate sss onto Model Grids
#Still need to convert to potential Salt
#Shifts Lon values to match model and then interpolates

salt_obs_file ='/g/data1/v45/pas561/mom/archive/WOA/woa13_decav_s00_04v2.nc'

x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)

# Extract the variables
nc = Dataset(salt_obs_file, mode='r') # file handle, open in read only mode
lon = nc.variables['lon'][:]
lat = nc.variables['lat'][:]
zt = nc.variables['depth'][:]
s_an = nc.variables['s_an'][:]
#mld = nc.variables['mld'][:]
#ty_trans_rho = nc.variables['ty_trans_rho'][0,:,:,:]
nc.close() # close the file
print s_an.shape

clev = np.arange(28,36,.05)
```

```
plt.subplot(1,2,1)
plt.contourf(lon,lat,s_an[0,0,:,:],levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.xlabel('Longitude ($^\circ$W)')
plt.ylabel('Latitude ($^\circ$N)')
plt.title('WOA In Situ Salt')

#re-arrange lon values (-180:180) to match xt_ocean (-280:80)
lon[lon>80] -= 360
lon_copy = np.ma.copy(lon)
shift_index0 = np.argmin(lon)
shift_index1 = len(lon) - shift_index0
lon[:shift_index1] = lon_copy[shift_index0:]
lon[shift_index1:] = lon_copy[:shift_index0]

s_an_copy = np.ma.copy(s_an)
s_an[:,:,:,:shift_index1] = s_an_copy[:,:,:,shift_index0:]
s_an[:,:,:,shift_index1:] = s_an_copy[:,:,:,:shift_index0]
#s_an_c = np.ma.masked_invalid(s_an)
print s_an.shape

sss=s_an[0,0,:,:]
print sss.shape
print lon.shape
print lat.shape
print xt_ocean.shape
#convert lon and lat to 2d arrays
lon_2d,lat_2d = np.meshgrid(lon,lat)
#amps_u_interp = scipy.interpolate.griddata((lon.flatten(),lat.flatten()),sss.
↪flatten(),(xt_2d,yt_2d),method='linear')
woa_sss_interp = scipy.interpolate.griddata((lon_2d.flatten(),lat_2d.flatten()),sss.
↪flatten(),(geolon_t,geolat_t),method='linear')
# I found I needed this to mask a bunch of NaNs after:
woa_sss_interp = np.ma.masked_invalid(woa_sss_interp)
print woa_sss_interp.shape

plt.subplot(1,2,2)
plt.contourf(geolon_t,geolat_t,woa_sss_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.xlabel('Longitude ($^\circ$W)')
plt.ylabel('Latitude ($^\circ$N)')
plt.title('WOA Salt on MOM01 Grid')
```

```
(1, 102, 720, 1440)
(1, 102, 720, 1440)
(720, 1440)
(1440,)
(720,)
(3600,)
(2700, 3600)
```

```
<matplotlib.text.Text at 0x7fad4c4f7b50>
```

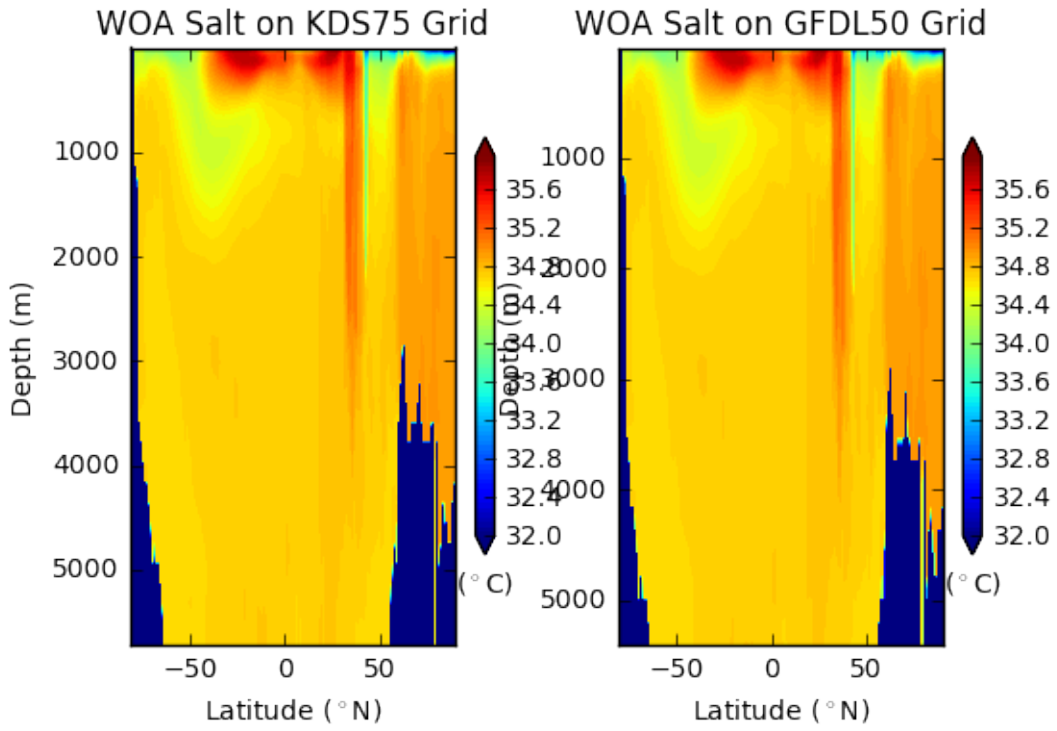



```
#Get WOA observed Salt (In Situ) data and calculate zonal average
#interpolate zonal average onto MOM grids - 50 and 75 vertical levels
#Still need to convert from in situ to potential Salt

x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)

zavgt=np.ma.mean(s_an[0,:,:,:],axis=2)
print zavgt.shape
print zt.shape
print lat.shape
print st75_ocean.shape
print st50_ocean.shape


########## for 1D lat/long coords, can change degree of kx for improved fit if needed:
from scipy.interpolate import RectBivariateSpline
f = RectBivariateSpline(zt,lat,zavgt,kx=1,ky=1)
woa_zavg75_interp = f(st75_ocean,yt_ocean)
# I found I needed this to mask a bunch of NaNs after:
woa_zavg75_interp  = np.ma.masked_invalid(woa_zavg75_interp)
print woa_zavg75_interp.shape

clev = np.arange(32,36,.05)

plt.subplot(1,2,1)
plt.contourf(yt_ocean,st75_ocean,woa_zavg75_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.gca().invert_yaxis()
plt.ylabel('Depth (m)')
plt.xlabel('Latitude ($^\circ$N)')
plt.title('WOA Salt on KDS75 Grid')
```

```
########## for 1D lat/long coords, can change degree of kx for improved fit if needed:
from scipy.interpolate import RectBivariateSpline
f = RectBivariateSpline(zt,lat,zavgt,kx=1,ky=1)
woa_zavg50_interp = f(st50_ocean,yt_ocean)
# I found I needed this to mask a bunch of NaNs after:
woa_zavg50_interp  = np.ma.masked_invalid(woa_zavg50_interp)
print woa_zavg50_interp.shape

plt.subplot(1,2,2)
plt.contourf(yt_ocean,st50_ocean,woa_zavg50_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.gca().invert_yaxis()
plt.ylabel('Depth (m)')
plt.xlabel('Latitude ($^\circ$N)')
plt.title('WOA Salt on GFDL50 Grid')
```

```
(102, 720)
(102,)
(720,)
(75,)
(50,)
(75, 2700)
(50, 2700)
```

```
<matplotlib.text.Text at 0x7fad4e568fd0>
```




```
DataDir = '/g/data3/hh5/tmp/cosima/mom01v5'
```

```
expts = ['GFDL50','KDS75','KDS75_UP','KDS75_PI']
#expts = ['GFDL50','KDS75']
```

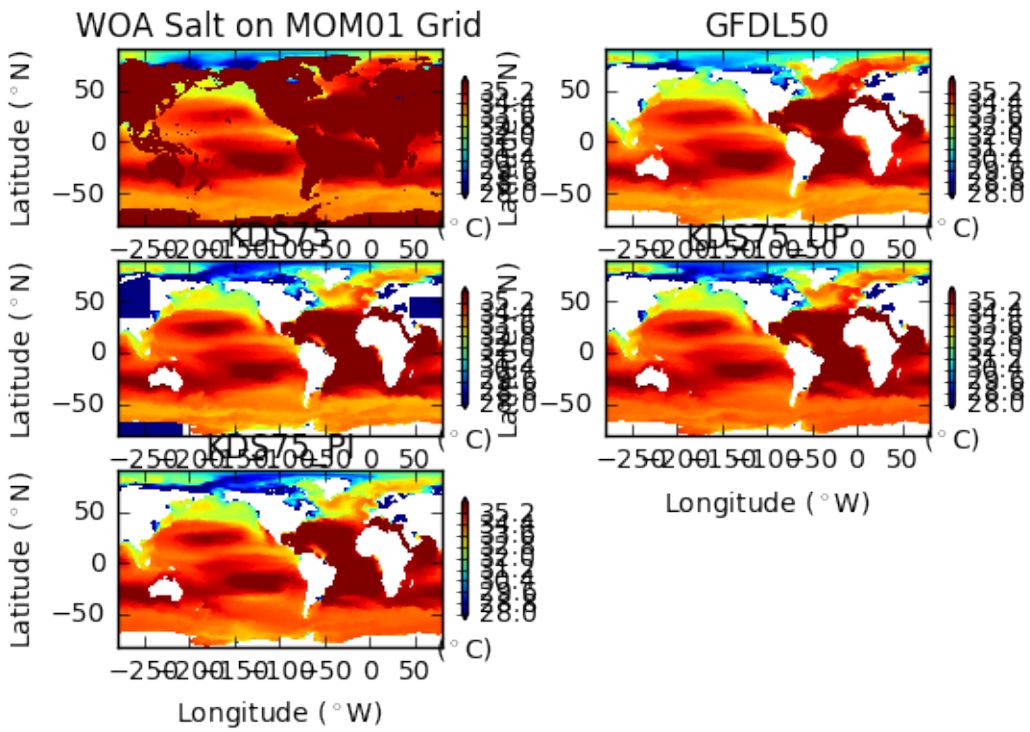
```
#calculate annual mean surface and zonal average salt from last year of MOM runs

#should probably do this with lists? but takes forever to convert to numpy array
sss=np.zeros((len(expts),yt_ocean.shape[0],xt_ocean.shape[0]))
zavg75=np.zeros((len(expts)-1,st75_ocean.shape[0],yt_ocean.shape[0]))
zavg50=np.zeros((1,st50_ocean.shape[0],yt_ocean.shape[0]))

ii=0
for e in expts:

    ExpDir = os.path.join(DataDir,e)

    if e=="GFDL50":
        OceanFile = os.path.join(DataDir,e,'output4*/ocean.nc')
        avgt=4
    if e=="KDS75":
        OceanFile = os.path.join(DataDir,e,'output4*/ocean_month.nc')
        avgt=12
    if e=="KDS75_UP":
        OceanFile = os.path.join(DataDir,e,'output3*/ocean.nc')
        avgt=4
    if e=="KDS75_PI":
        OceanFile = os.path.join(DataDir,e,'output3*/ocean.nc')
        avgt=4

    print OceanFile
    ff=xarray.open_mfdataset(OceanFile,engine='netcdf4',concat_dim='time',decode_
↪times=False,
                             chunks={'time':1,'st_ocean':5,'xt_ocean':338,'yt_ocean
↪':450})

    tavg=ff.salt[-avgt:,1,:,:].mean('time').load()
    print tavg.shape
    sss[ii,:,:]=tavg

    if ff.st_ocean.shape[0]==50:
        zavg50[0,:,:]=ff.salt[-avgt:,:,:,:].mean('time').mean('xt_ocean').load()
        print ff.st_ocean.shape[0]
    if ff.st_ocean.shape[0]==75:
        zavg75[ii-1,:,:]=ff.salt[-avgt:,:,:,:].mean('time').mean('xt_ocean').load()
        print ff.st_ocean.shape[0]

    ii+=1
```

```
/g/data3/hh5/tmp/cosima/mom01v5/GFDL50/output4*/ocean.nc
(2700, 3600)
50
/g/data3/hh5/tmp/cosima/mom01v5/KDS75/output4*/ocean_month.nc
(2700, 3600)
75
/g/data3/hh5/tmp/cosima/mom01v5/KDS75_UP/output3*/ocean.nc
(2700, 3600)
75
/g/data3/hh5/tmp/cosima/mom01v5/KDS75_PI/output3*/ocean.nc
```

```
(2700, 3600)
75
```

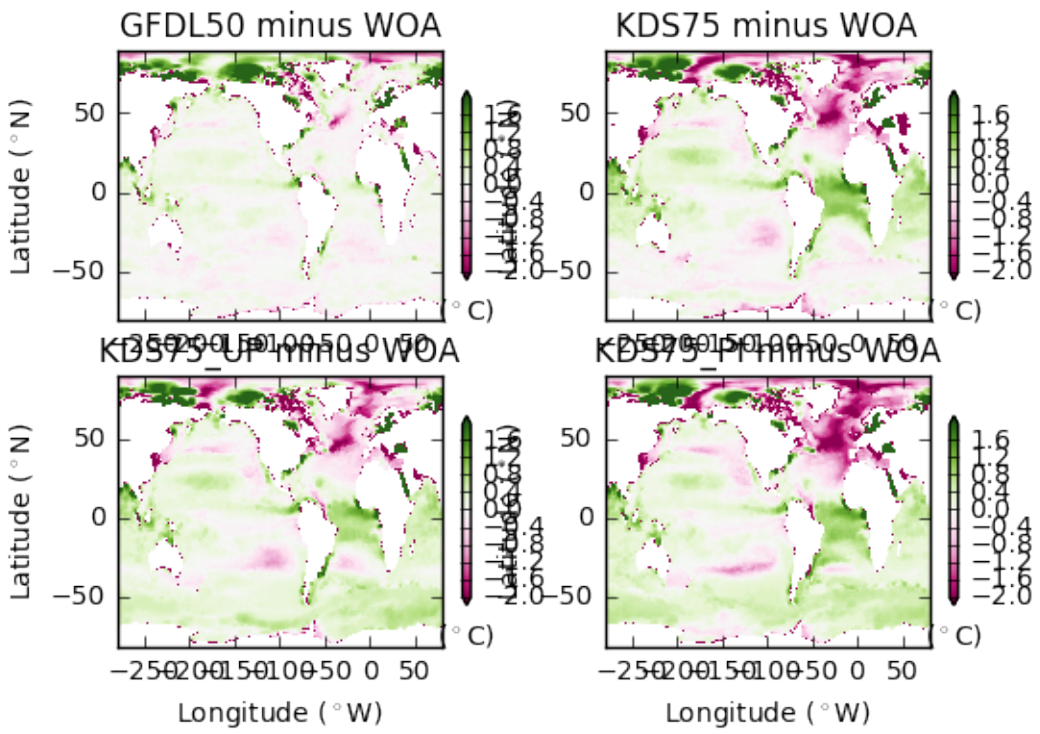
```
#plot annual mean surface salt from last year of MOM runs

clev = np.arange(28,36,.05)

plt.subplot(3,2,1)
plt.contourf(geolon_t,geolat_t,woa_sss_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.xlabel('Longitude ($^\circ$W)')
plt.ylabel('Latitude ($^\circ$N)')
plt.title('WOA Salt on MOM01 Grid')

ii=0
for e in expts:

    #plot full sss
    plt.subplot(3,2,ii+2)
    plt.contourf(geolon_t,geolat_t,sss[ii,:,:],levels=clev,extend='both')
    cb=plt.colorbar(orientation='vertical',shrink = 0.7)
    cb.ax.set_xlabel('($^\circ$C)')
    plt.xlabel('Longitude ($^\circ$W)')
    plt.ylabel('Latitude ($^\circ$N)')
    plt.title(e)

    ii+=1
    print e, ii

plt.savefig('mom01_sss.pdf')
```

```
GFDL50 1
KDS75 2
KDS75_UP 3
KDS75_PI 4
```




```
#plot sss anomalies relative to WOA obs data on model grid

clev = np.arange(-2,2,.05)

ii=0
for e in expts:

    #plot full sss
    plt.subplot(2,2,ii+1)
    plt.contourf(geolon_t,geolat_t,sss[ii,:,:]-woa_sss_interp,cmap=plt.cm.PiYG,
↪levels=clev,extend='both')
    cb=plt.colorbar(orientation='vertical',shrink = 0.7)
    cb.ax.set_xlabel('($^\circ$C)')
    plt.xlabel('Longitude ($^\circ$W)')
    plt.ylabel('Latitude ($^\circ$N)')
    plt.title(e+ " minus WOA")

    ii+=1
    print e, ii

plt.savefig('mom01_sssanom.pdf')
```

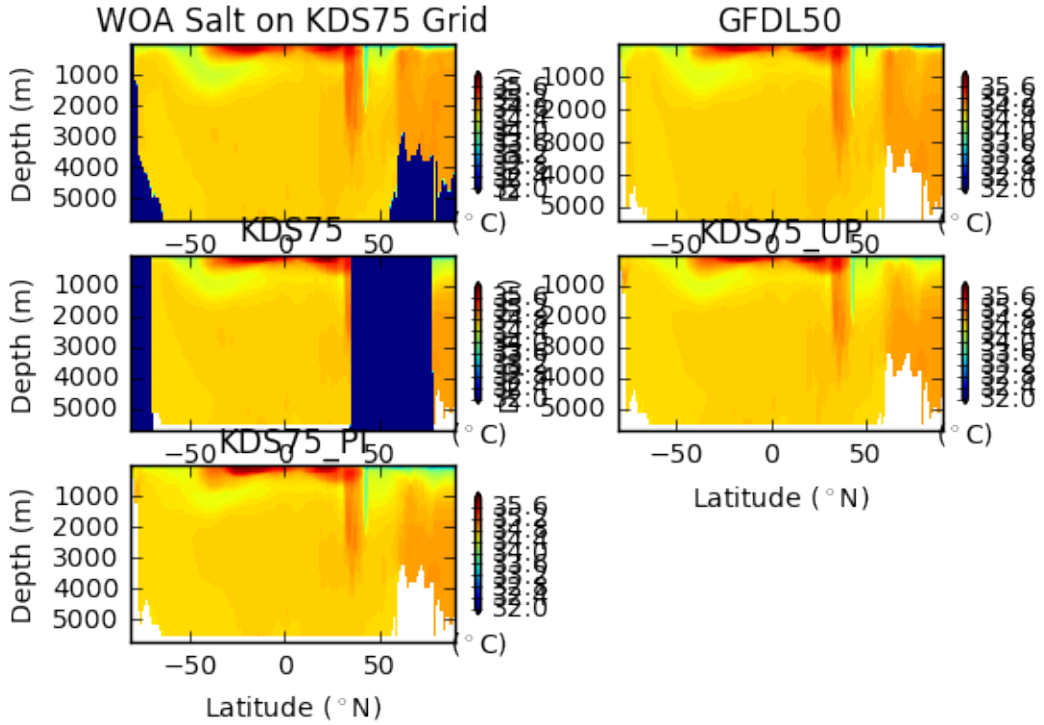
```
GFDL50 1
KDS75 2
KDS75_UP 3
KDS75_PI 4
```




```
#plot annual mean zonal average salt from last year of MOM runs and WOA
#

clev = np.arange(32,36,.05)

plt.subplot(3,2,1)
plt.contourf(yt_ocean,ff.st_ocean,woa_zavg75_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.gca().invert_yaxis()
plt.ylabel('Depth (m)')
plt.xlabel('Latitude ($^\circ$N)')
plt.title('WOA Salt on KDS75 Grid')


ii=0
for e in expts:

    if e=="GFDL50":
        plt.subplot(3,2,ii+2)
        plt.contourf(yt_ocean,st50_ocean,zavg50[ii,:,:],levels=clev,extend='both')
        cb=plt.colorbar(orientation='vertical',shrink = 0.7)
        cb.ax.set_xlabel('($^\circ$C)')
        plt.gca().invert_yaxis()
        plt.ylabel('Depth (m)')
        plt.xlabel('Latitude ($^\circ$N)')
        plt.title(e)

    else:
        plt.subplot(3,2,ii+2)
        plt.contourf(yt_ocean,st75_ocean,zavg75[ii-1,:,:],levels=clev,extend='both')
        cb=plt.colorbar(orientation='vertical',shrink = 0.7)
```

```
        cb.ax.set_xlabel('($^\circ$C)')
        plt.gca().invert_yaxis()
        plt.ylabel('Depth (m)')
        plt.xlabel('Latitude ($^\circ$N)')
        plt.title(e)

    ii+=1
    print e, ii
plt.savefig('mom01_zavS.pdf')


#it looks like the land cell masking is screwing up the zavg in KDS75
```

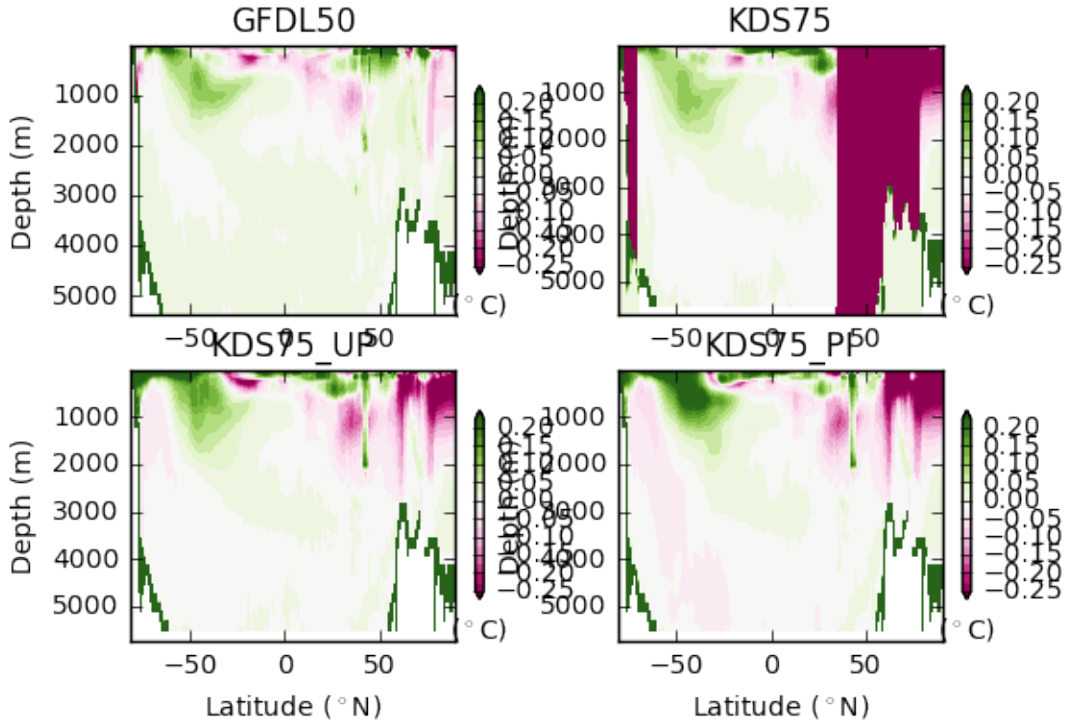
```
GFDL50 1
KDS75 2
KDS75_UP 3
KDS75_PI 4
```




```
clev = np.arange(-.25,.25,.025)

#plot zonal average anomaly relative to WOA
ii=0
for e in expts:

    if e=="GFDL50":
        plt.subplot(2,2,ii+1)
        plt.contourf(yt_ocean,st50_ocean,zavg50[ii,:,:]-woa_zavg50_interp,cmap=plt.cm.
↪PiYG,levels=clev,extend='both')
        cb=plt.colorbar(orientation='vertical',shrink = 0.7)
        cb.ax.set_xlabel('($^\circ$C)')
        plt.gca().invert_yaxis()
```

```
        plt.ylabel('Depth (m)')
        plt.xlabel('Latitude ($^\circ$N)')
        plt.title(e)

    else:
        plt.subplot(2,2,ii+1)
        plt.contourf(yt_ocean,st75_ocean,zavg75[ii-1,:,:]-woa_zavg75_interp,cmap=plt.
↪cm.PiYG,levels=clev,extend='both')
        cb=plt.colorbar(orientation='vertical',shrink = 0.7)
        cb.ax.set_xlabel('($^\circ$C)')
        plt.gca().invert_yaxis()
        plt.ylabel('Depth (m)')
        plt.xlabel('Latitude ($^\circ$N)')
        plt.title(e)

    ii+=1
    print e, ii
#it looks like the land cell masking is screwing the zavg up in KDS75
plt.savefig('mom01_zavgSanom.pdf')
```

```
GFDL50 1
KDS75 2
KDS75_UP 3
KDS75_PI 4
```



MOM01 Temp Evaluations

Surface and zonal average temps across simulations

computes anomalies relative to longterm average of WOA data takes forever to run thru all expts, I suggest trying a subset

expts = ['GFDL50','KDS75']

still need to convert WOA insitu temp to potential temp: not sure how to vectorize this.

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
# Load modules
    from netCDF4 import Dataset # to work with NetCDF files
import numpy as np
import matplotlib.pyplot as plt # to generate plots
from mpl_toolkits.basemap import Basemap # plot on map projections
from os.path import expanduser
home = expanduser("~") # Get users home directory
import os # operating system interface

import xarray
from xarray.ufuncs import square, sqrt
import scipy.interpolate

#import scipy
#sfrom scipy.interpolate import RegularGridInterpolator, griddata

#import dask
#from dask import delayed
#import dask.array as da
#import dask.dataframe as dd
#from dask.multiprocessing import get
#from dask.async import get_sync
#from dask.diagnostics import ProgressBar

#import pandas as pd
#from glob import glob
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
#pbar = ProgressBar()
#pbar.register()
```

```
# set up dask to initiate 4 parallel processors.
#this doesn't work yet
#dask.set_options(get=dask.multiprocessing.get, num_workers=4)
```

```
#extract the MOM model grid info - 50 and 75 vertical levels
hgrid_file ='/g/data1/v45/pas561/mom/archive/VertOverturn/mom01_unmasked_ocean_grid.nc
↪'
vgrid75_file ='/g/data1/v45/pas561/mom/archive/VertOverturn/kds75.uvwt.230-257.ncra.nc
↪'
vgrid50_file ='/g/data3/hh5/tmp/cosima/mom01v5/GFDL50/gfdl50.uvwt.384_403.ncra.nc'

# Extract the variables
```

```
nc = Dataset(hgrid_file, mode='r') # file handle, open in read only mode
geolon_t = nc.variables['geolon_t'][:]
geolat_t = nc.variables['geolat_t'][:]
area_t = nc.variables['area_t'][:]
dxt = nc.variables['dxt'][:]
dyt = nc.variables['dyt'][:]
nc.close() # close the file
print geolon_t.shape

nc = Dataset(vgrid75_file, mode='r') # file handle, open in read only mode
sw75_ocean = nc.variables['sw_ocean'][:]
st75_ocean = nc.variables['st_ocean'][:]
yt_ocean = nc.variables['yt_ocean'][:]
xt_ocean = nc.variables['xt_ocean'][:]
nc.close() # close the file

nc = Dataset(vgrid50_file, mode='r') # file handle, open in read only mode
sw50_ocean = nc.variables['sw_ocean'][:]
st50_ocean = nc.variables['st_ocean'][:]
yt_ocean = nc.variables['yt_ocean'][:]
nc.close() # close the file

print st75_ocean.shape
print st50_ocean.shape
```

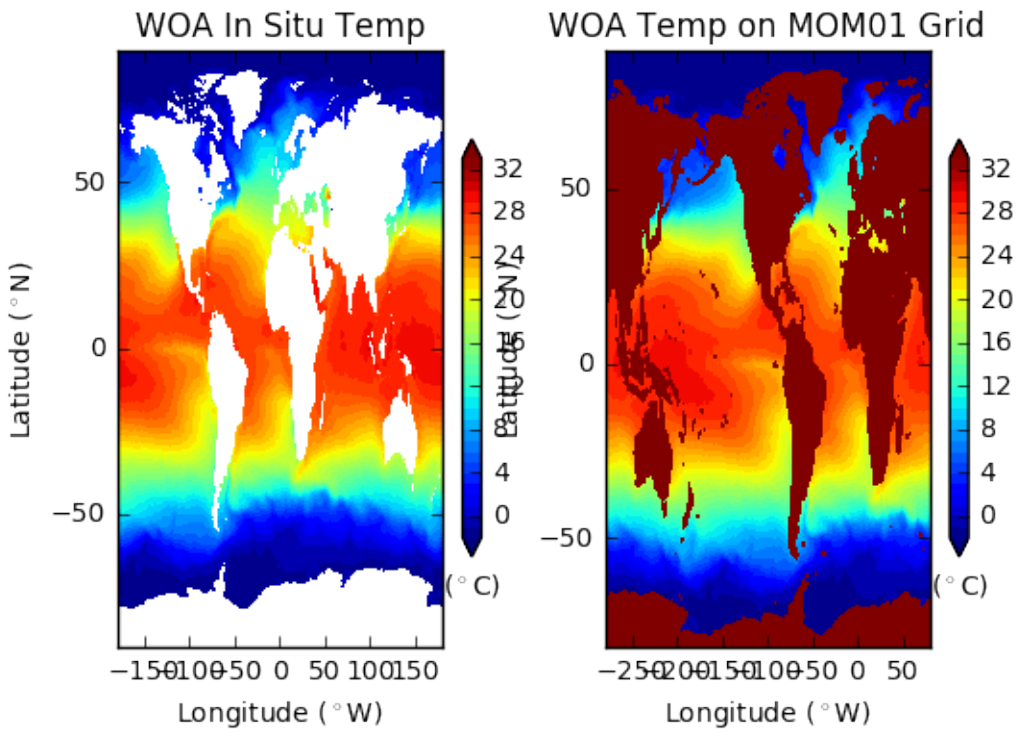
```
(2700, 3600)
(75,)
(50,)
```

```
#Get WOA observed Temp (In Situ) data and interpolate SST onto Model Grids
#Still need to convert to potential Temp
#Shifts Lon values to match model and then interpolates

temp_obs_file ='/g/data1/v45/pas561/mom/archive/WOA/woa13_decav_t00_04v2.nc'

x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)

# Extract the variables
nc = Dataset(temp_obs_file, mode='r') # file handle, open in read only mode
lon = nc.variables['lon'][:]
lat = nc.variables['lat'][:]
zt = nc.variables['depth'][:]
t_an = nc.variables['t_an'][:]
#mld = nc.variables['mld'][:]
#ty_trans_rho = nc.variables['ty_trans_rho'][0,:,:,:]
nc.close() # close the file
print t_an.shape

plt.subplot(1,2,1)
clev = np.arange(-2,34,1)
plt.contourf(lon,lat,t_an[0,0,:,:],levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.xlabel('Longitude ($^\circ$W)')
plt.ylabel('Latitude ($^\circ$N)')
plt.title('WOA In Situ Temp')
```

```
#re-arrange lon values (-180:180) to match xt_ocean (-280:80)
lon[lon>80] -= 360
lon_copy = np.ma.copy(lon)
shift_index0 = np.argmin(lon)
shift_index1 = len(lon) - shift_index0
lon[:shift_index1] = lon_copy[shift_index0:]
lon[shift_index1:] = lon_copy[:shift_index0]

t_an_copy = np.ma.copy(t_an)
t_an[:,:,:,:shift_index1] = t_an_copy[:,:,:,shift_index0:]
t_an[:,:,:,shift_index1:] = t_an_copy[:,:,:,:shift_index0]
#t_an_c = np.ma.masked_invalid(t_an)
print t_an.shape

sst=t_an[0,0,:,:]
print sst.shape
print lon.shape
print lat.shape
print xt_ocean.shape
#convert lon and lat to 2d arrays
lon_2d,lat_2d = np.meshgrid(lon,lat)
#amps_u_interp = scipy.interpolate.griddata((lon.flatten(),lat.flatten()),sst.
↪flatten(),(xt_2d,yt_2d),method='linear')
woa_sst_interp = scipy.interpolate.griddata((lon_2d.flatten(),lat_2d.flatten()),sst.
↪flatten(),(geolon_t,geolat_t),method='linear')
# I found I needed this to mask a bunch of NaNs after:
woa_sst_interp = np.ma.masked_invalid(woa_sst_interp)
print woa_sst_interp.shape

plt.subplot(1,2,2)
clev = np.arange(-2,34,1)
plt.contourf(geolon_t,geolat_t,woa_sst_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.xlabel('Longitude ($^\circ$W)')
plt.ylabel('Latitude ($^\circ$N)')
plt.title('WOA Temp on MOM01 Grid')
```

```
(1, 102, 720, 1440)
(1, 102, 720, 1440)
(720, 1440)
(1440,)
(720,)
(3600,)
(2700, 3600)
```

```
<matplotlib.text.Text at 0x7f6cf5c59d50>
```

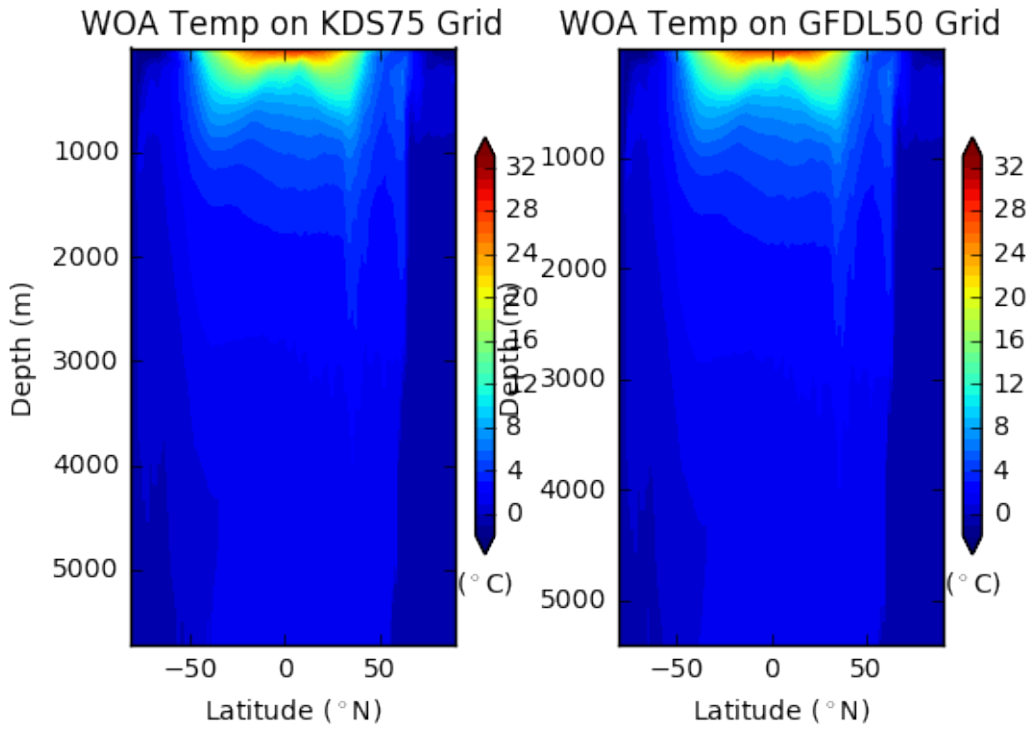



```
#Get WOA observed Temp (In Situ) data and calculate zonal average
#interpolate zonal average onto MOM grids - 50 and 75 vertical levels
#Still need to convert from in situ to potential Temp

x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)

zavgt=np.ma.mean(t_an[0,:,:,:],axis=2)
print zavgt.shape
print zt.shape
print lat.shape
print st75_ocean.shape
print st50_ocean.shape


########## for 1D lat/long coords, can change degree of kx for improved fit if needed:
from scipy.interpolate import RectBivariateSpline
f = RectBivariateSpline(zt,lat,zavgt,kx=1,ky=1)
woa_zavg75_interp = f(st75_ocean,yt_ocean)
# I found I needed this to mask a bunch of NaNs after:
woa_zavg75_interp  = np.ma.masked_invalid(woa_zavg75_interp)
print woa_zavg75_interp.shape

plt.subplot(1,2,1)
clev = np.arange(-2,34,1)
plt.contourf(yt_ocean,st75_ocean,woa_zavg75_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.gca().invert_yaxis()
plt.ylabel('Depth (m)')
plt.xlabel('Latitude ($^\circ$N)')
plt.title('WOA Temp on KDS75 Grid')
```

```
########## for 1D lat/long coords, can change degree of kx for improved fit if needed:
from scipy.interpolate import RectBivariateSpline
f = RectBivariateSpline(zt,lat,zavgt,kx=1,ky=1)
woa_zavg50_interp = f(st50_ocean,yt_ocean)
# I found I needed this to mask a bunch of NaNs after:
woa_zavg50_interp  = np.ma.masked_invalid(woa_zavg50_interp)
print woa_zavg50_interp.shape

plt.subplot(1,2,2)
clev = np.arange(-2,34,1)
plt.contourf(yt_ocean,st50_ocean,woa_zavg50_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.gca().invert_yaxis()
plt.ylabel('Depth (m)')
plt.xlabel('Latitude ($^\circ$N)')
plt.title('WOA Temp on GFDL50 Grid')
```

```
(102, 720)
(102,)
(720,)
(75,)
(50,)
(75, 2700)
(50, 2700)
```

```
<matplotlib.text.Text at 0x7f6c8b0d9a90>
```




```
DataDir = '/g/data3/hh5/tmp/cosima/mom01v5'
```

```
expts = ['GFDL50','KDS75','KDS75_UP','KDS75_PI']
#expts = ['GFDL50','KDS75']
```

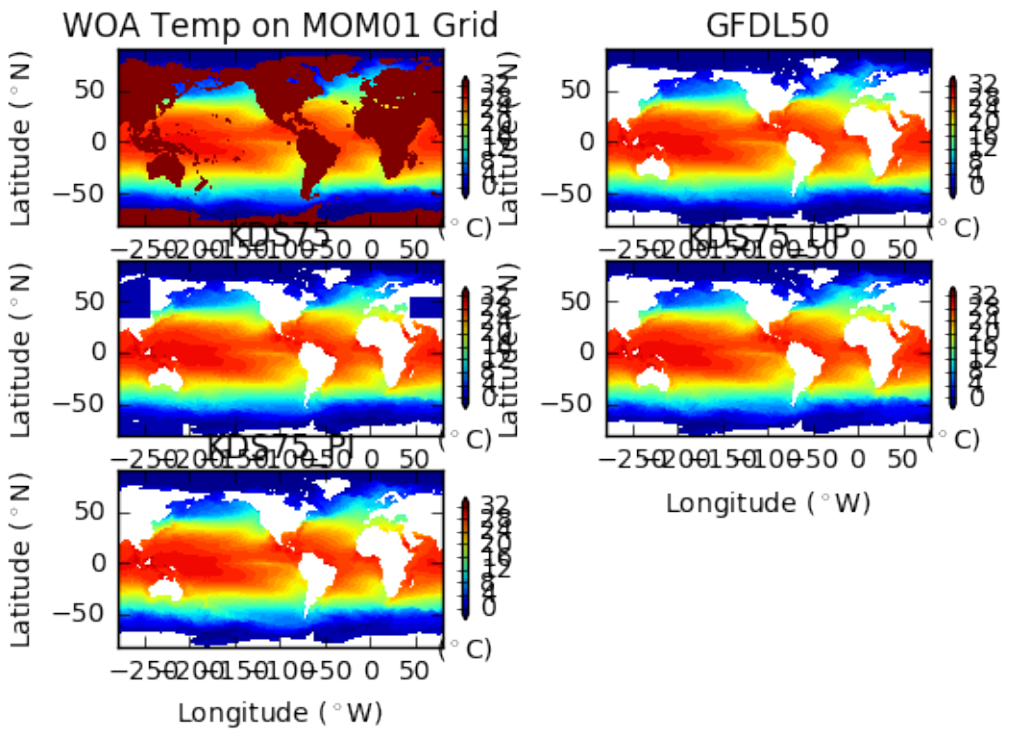
```
#calculate annual mean surface and zonal average temp from last year of MOM runs

#should probably do this with lists? but takes forever to convert to numpy array
sst=np.zeros((len(expts),yt_ocean.shape[0],xt_ocean.shape[0]))
zavg75=np.zeros((len(expts)-1,st75_ocean.shape[0],yt_ocean.shape[0]))
zavg50=np.zeros((1,st50_ocean.shape[0],yt_ocean.shape[0]))

ii=0
for e in expts:

    ExpDir = os.path.join(DataDir,e)

    if e=="GFDL50":
        OceanFile = os.path.join(DataDir,e,'output4*/ocean.nc')
        avgt=4
    if e=="KDS75":
        OceanFile = os.path.join(DataDir,e,'output4*/ocean_month.nc')
        avgt=12
    if e=="KDS75_UP":
        OceanFile = os.path.join(DataDir,e,'output3*/ocean.nc')
        avgt=4
    if e=="KDS75_PI":
        OceanFile = os.path.join(DataDir,e,'output3*/ocean.nc')
        avgt=4

    print OceanFile
    ff=xarray.open_mfdataset(OceanFile,engine='netcdf4',concat_dim='time',decode_
↪times=False,
                             chunks={'time':1,'st_ocean':5,'xt_ocean':338,'yt_ocean
↪':450})

    tavg=ff.temp[-avgt:,1,:,:].mean('time').load()
    print tavg.shape
    sst[ii,:,:]=tavg

    if ff.st_ocean.shape[0]==50:
        zavg50[0,:,:]=ff.temp[-avgt:,:,:,:].mean('time').mean('xt_ocean').load()
        print ff.st_ocean.shape[0]
    if ff.st_ocean.shape[0]==75:
        zavg75[ii-1,:,:]=ff.temp[-avgt:,:,:,:].mean('time').mean('xt_ocean').load()
        print ff.st_ocean.shape[0]

    ii+=1
```

```
/g/data3/hh5/tmp/cosima/mom01v5/GFDL50/output4*/ocean.nc
(2700, 3600)
50
/g/data3/hh5/tmp/cosima/mom01v5/KDS75/output4*/ocean_month.nc
(2700, 3600)
75
/g/data3/hh5/tmp/cosima/mom01v5/KDS75_UP/output3*/ocean.nc
(2700, 3600)
75
/g/data3/hh5/tmp/cosima/mom01v5/KDS75_PI/output3*/ocean.nc
```

```
(2700, 3600)
75
```

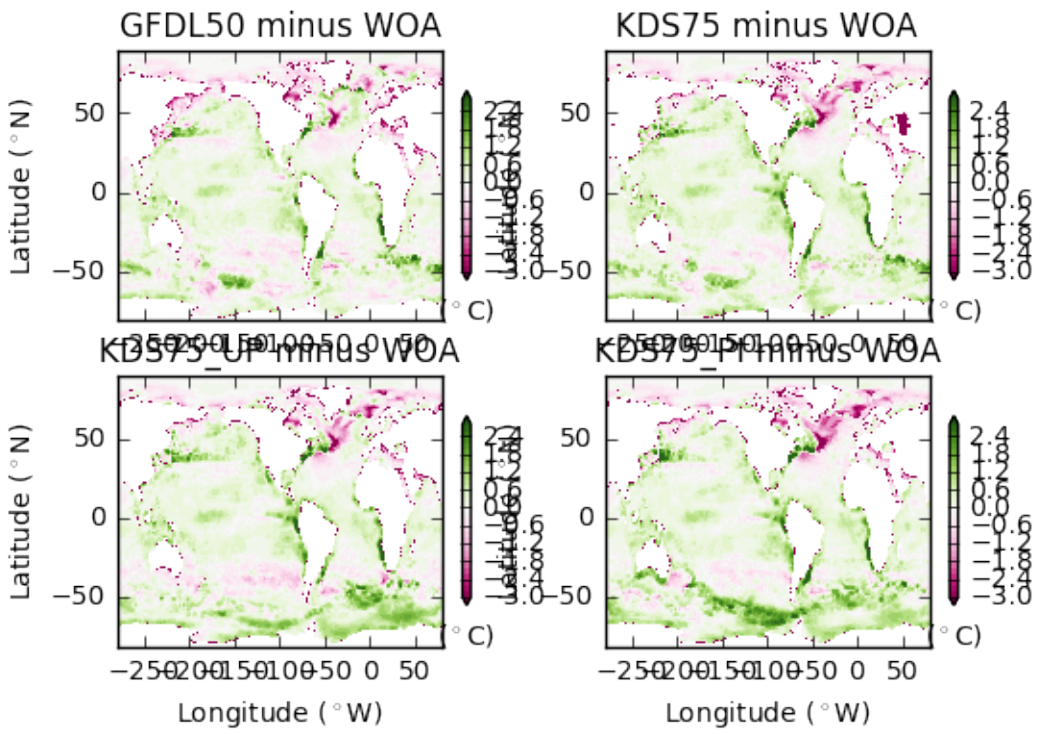
```
#plot annual mean surface temp from last year of MOM runs

plt.subplot(3,2,1)
clev = np.arange(-2,34,1)
plt.contourf(geolon_t,geolat_t,woa_sst_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.xlabel('Longitude ($^\circ$W)')
plt.ylabel('Latitude ($^\circ$N)')
plt.title('WOA Temp on MOM01 Grid')

ii=0
for e in expts:

    #plot full SST
    plt.subplot(3,2,ii+2)
    clev = np.arange(-2,34,1)
    plt.contourf(geolon_t,geolat_t,sst[ii,:,:],levels=clev,extend='both')
    cb=plt.colorbar(orientation='vertical',shrink = 0.7)
    cb.ax.set_xlabel('($^\circ$C)')
    plt.xlabel('Longitude ($^\circ$W)')
    plt.ylabel('Latitude ($^\circ$N)')
    plt.title(e)

    ii+=1
    print ii,e

plt.savefig('mom01_sst.pdf')
```

```
1 GFDL50
2 KDS75
3 KDS75_UP
4 KDS75_PI
```



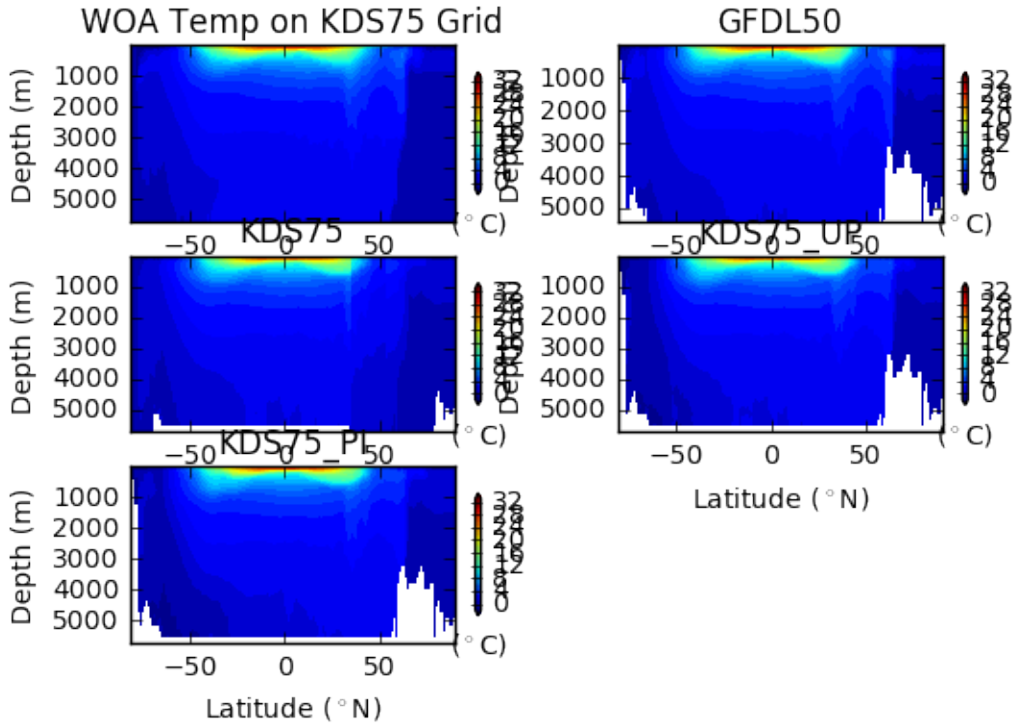

```
#plot SST anomalies relative to WOA obs data on model grid

ii=0
for e in expts:

    #plot full SST
    plt.subplot(2,2,ii+1)
    clev = np.arange(-3,3,.2)
    plt.contourf(geolon_t,geolat_t,sst[ii,:,:]-woa_sst_interp,cmap=plt.cm.PiYG,
↪levels=clev,extend='both')
    cb=plt.colorbar(orientation='vertical',shrink = 0.7)
    cb.ax.set_xlabel('($^\circ$C)')
    plt.xlabel('Longitude ($^\circ$W)')
    plt.ylabel('Latitude ($^\circ$N)')
    plt.title(e+ " minus WOA")

    ii+=1
    print e

plt.savefig('mom01_sstanom.pdf')
```

```
GFDL50
KDS75
KDS75_UP
KDS75_PI
```



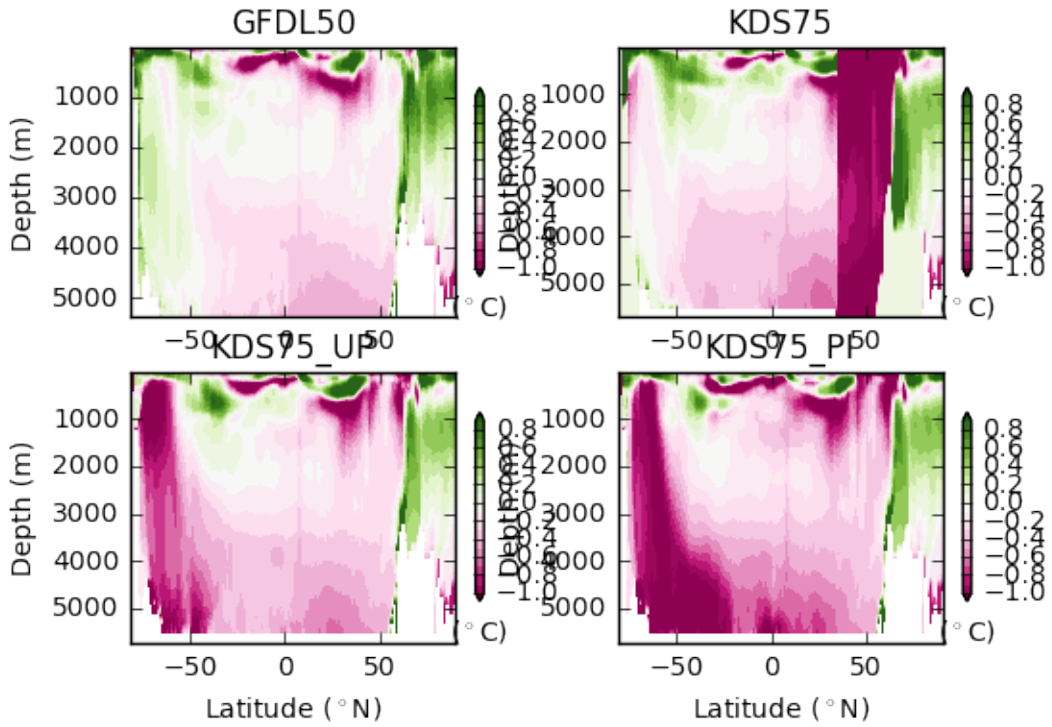

```
#plot annual mean zonal average temp from last year of MOM runs and WOA
plt.subplot(3,2,1)
clev = np.arange(-2,34,1)
plt.contourf(yt_ocean,ff.st_ocean,woa_zavg75_interp,levels=clev,extend='both')
cb=plt.colorbar(orientation='vertical',shrink = 0.7)
cb.ax.set_xlabel('($^\circ$C)')
plt.gca().invert_yaxis()
plt.ylabel('Depth (m)')
plt.xlabel('Latitude ($^\circ$N)')
plt.title('WOA Temp on KDS75 Grid')


ii=0
for e in expts:

    if e=="GFDL50":
        plt.subplot(3,2,ii+2)
        clev = np.arange(-2,34,1)
        plt.contourf(yt_ocean,st50_ocean,zavg50[ii,:,:],levels=clev,extend='both')
        cb=plt.colorbar(orientation='vertical',shrink = 0.7)
        cb.ax.set_xlabel('($^\circ$C)')
        plt.gca().invert_yaxis()
        plt.ylabel('Depth (m)')
        plt.xlabel('Latitude ($^\circ$N)')
        plt.title(e)

    else:
        plt.subplot(3,2,ii+2)
        clev = np.arange(-2,34,1)
        plt.contourf(yt_ocean,st75_ocean,zavg75[ii-1,:,:],levels=clev,extend='both')
        cb=plt.colorbar(orientation='vertical',shrink = 0.7)
        cb.ax.set_xlabel('($^\circ$C)')
```

```
        plt.gca().invert_yaxis()
        plt.ylabel('Depth (m)')
        plt.xlabel('Latitude ($^\circ$N)')
        plt.title(e)

    ii+=1
plt.savefig('mom01_zavT.pdf')
```




```
#plot zonal average anomaly relative to WOA
ii=0
for e in expts:

    if e=="GFDL50":
        plt.subplot(2,2,ii+1)
        clev = np.arange(-1,1,.1)
        plt.contourf(yt_ocean,st50_ocean,zavg50[ii,:,:]-woa_zavg50_interp,cmap=plt.cm.
↪PiYG,levels=clev,extend='both')
        cb=plt.colorbar(orientation='vertical',shrink = 0.7)
        cb.ax.set_xlabel('($^\circ$C)')
        plt.gca().invert_yaxis()
        plt.ylabel('Depth (m)')
        plt.xlabel('Latitude ($^\circ$N)')
        plt.title(e)

    else:
        plt.subplot(2,2,ii+1)
        clev = np.arange(-1,1,.1)
        plt.contourf(yt_ocean,st75_ocean,zavg75[ii-1,:,:]-woa_zavg75_interp,cmap=plt.
↪cm.PiYG,levels=clev,extend='both')
        cb=plt.colorbar(orientation='vertical',shrink = 0.7)
        cb.ax.set_xlabel('($^\circ$C)')
```

```
        plt.gca().invert_yaxis()
        plt.ylabel('Depth (m)')
        plt.xlabel('Latitude ($^\circ$N)')
        plt.title(e)

    ii+=1

#plt.savefig('mom01_zavgTanom.pdf')
```



Ice Diagnostics

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
from glob import glob
import os
import xarray as xr
import cartopy
import matplotlib.pyplot as plt
import numpy as np
import cartopy.crs as ccrs
import cartopy.feature
    from IPython.display import HTML
```

```
from dask.diagnostics import ProgressBar
pbar = ProgressBar()
pbar.register()
```

```
DataDir = '/g/data3/hh5/tmp/cosima/mom01v5'
expt = 'KDS75'
```

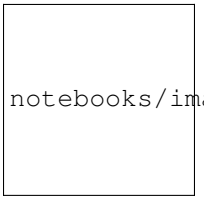
```
expdir = os.path.join(DataDir, expt)
dataFileList = glob(os.path.join(expdir, 'output*/ice_month.nc'))
dataFileList.sort()
```

```
gridFileList = glob(os.path.join(expdir, 'output*/ocean_grid.nc'))
gridFileList.sort()
```

```
dsx_grid = xr.open_dataset(gridFileList[0], decode_times=False)
area_t = dsx_grid.area_t
```

```
dsx_grid.area_t.plot()
```

```
<matplotlib.collections.QuadMesh at 0x7f7f6d0cec18>
```



From `diag_table`

Saved in `ice_month.nc`

CN, HI, HS averaged globally over each month

```
dsx = xr.open_mfdataset(dataFileList[:12], decode_times=False, concat_dim='time')
```

There is an outstanding issue with year outside of a given range. Artificially shift the years to years since 2000.

```
dsx.time.attrs['units'] = 'days since 2000-01-01 00:00:00'
dsx.average_T1.attrs['units'] = 'days since 2000-01-01 00:00:00'
dsx.average_T2.attrs['units'] = 'days since 2000-01-01 00:00:00'
dsx = xr.decode_cf(dsx)
```

Annual ice volume

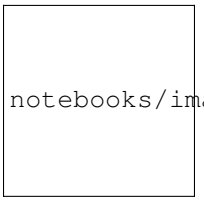
```
HI = dsx.sel(yt=slice(-90, -60)).HI
south_area_t = area_t.sel(yt_ocean=slice(-90,-60))
HI
```

```
<xarray.DataArray 'HI' (time: 72, yt: 490, xt: 3600)>
[127008000 values with dtype=float64]
Coordinates:
  * xt       (xt) float64 -280.0 -279.9 -279.8 -279.7 -279.6 -279.5 -279.4 ...
  * yt       (yt) float64 -81.11 -81.07 -81.02 -80.98 -80.94 -80.9 -80.86 ...
  * time     (time) datetime64[ns] 2010-02-15 2010-03-16T12:00:00 2010-04-16 ...
Attributes:
    long_name: ice thickness
    units: m-ice
    cell_methods: time: mean
    time_avg_info: average_T1,average_T2,average_DT
```

```
volume = south_area_t.values * HI
volume.name = 'Volume'
```

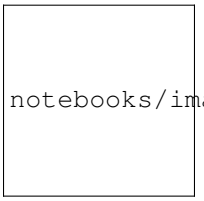
```
volume.sum(dim='yt').sum(dim='xt').plot(marker='o')
```

```
[<matplotlib.lines.Line2D at 0x7f7f6c706898>]
```




```
import matplotlib as mpl
from matplotlib import rc

rc('animation', ffmpeg_path='/short/v45/jm0634/conda/envs/my_analysis3/bin/ffmpeg')
rc('animation', html='html5')
```

```
plt.figure(figsize=[8,8])
ax1 = plt.subplot(1,1,1, projection=ccrs.SouthPolarStereo())
#ax1.set_extent([-180, 180, -90, -50], ccrs.PlateCarree())
ax1.add_feature(cartopy.feature.LAND)
#ax1.add_feature(cartopy.feature.OCEAN)
ax1.coastlines()
#ax1.stock_img()
ax1.gridlines()

quadmesh = dsx.HI.isel(time=0)\
            .sel(yt=slice(-90, -60))\
            .plot\
            .pcolormesh(ax=ax1,
                        transform=ccrs.PlateCarree(),
                        vmax=2)
```




```
import matplotlib.animation as animation

fig = plt.figure(figsize=[8,8])

ax1 = plt.subplot(1,1,1, projection=ccrs.SouthPolarStereo())
#ax1.set_extent([-180, 180, -90, -50], ccrs.PlateCarree())
ax1.add_feature(cartopy.feature.LAND)
#ax1.add_feature(cartopy.feature.OCEAN)
ax1.coastlines()
#ax1.stock_img()
ax1.gridlines()

quadmesh = dsx.HI.isel(time=0).sel(yt=slice(-90,
```

```
                                    -60)).plot(ax=ax1,
                                               transform=ccrs.PlateCarree(),
                                               vmax=2)
def animate(i):
    print(i)
    da = dsx.HI.isel(time=i).sel(yt=slice(-90,-60))
    quadmesh.set_array(da.to_masked_array().ravel())
    ax1.set_title('time = {}'.format(da.time.values))
    return quadmesh, ax1.title

def init_func():
    return quadmesh,
```


```
ani = animation.FuncAnimation(fig, animate, 6,
                              init_func=init_func,
                              interval=1000, blit=True, repeat_delay=3000)
```

```
ani
```

```
0
1
2
3
4
5
```

MOM-SIS 0.25° Diagnostics

This notebook calculates and retains key diagnostics from our mom01v5 simulations. The following experiments are included:

Experiment Name	Description
mom025_nyf	Original simulation, rerun from WOA13 initial conditions.
mom025_nyf_salt	New salt restoring file

Last updated May 20 2017.

Experiments

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
DataDir = '/g/data3/hh5/tmp/cosima/mom025'
expts = ['mom025_nyf']
```

```
import numpy as np
import matplotlib.pyplot as plt
from netCDF4 import Dataset
import os
import pickle

import dask
from dask import delayed
import dask.array as da
import dask.dataframe as dd
from dask.multiprocessing import get
from dask.async import get_sync
from dask.diagnostics import ProgressBar

import pandas as pd
import xarray as xr
from glob import glob
from mpl_toolkits.basemap import Basemap, shiftgrid

#import datashader
```

```
/g/data3/hh5/public/apps/miniconda3/envs/analysis3/lib/python3.5/site-packages/xarray/
↪core/formatting.py:16: FutureWarning: The pandas.tslib module is deprecated and␣
↪will be removed in a future version.
  from pandas.tslib import OutOfBoundsDatetime
```

```
pbar = ProgressBar()
pbar.register()
```

```
# set up dask to initiate 4 parallel processors.
dask.set_options(get=dask.multiprocessing.get, num_workers=4)
```

```
<dask.context.set_options at 0x7f603c96ec88>
```

Wind Stress fields

The following code block shows the zonal- and time-averaged wind stress forcing for each experiment.

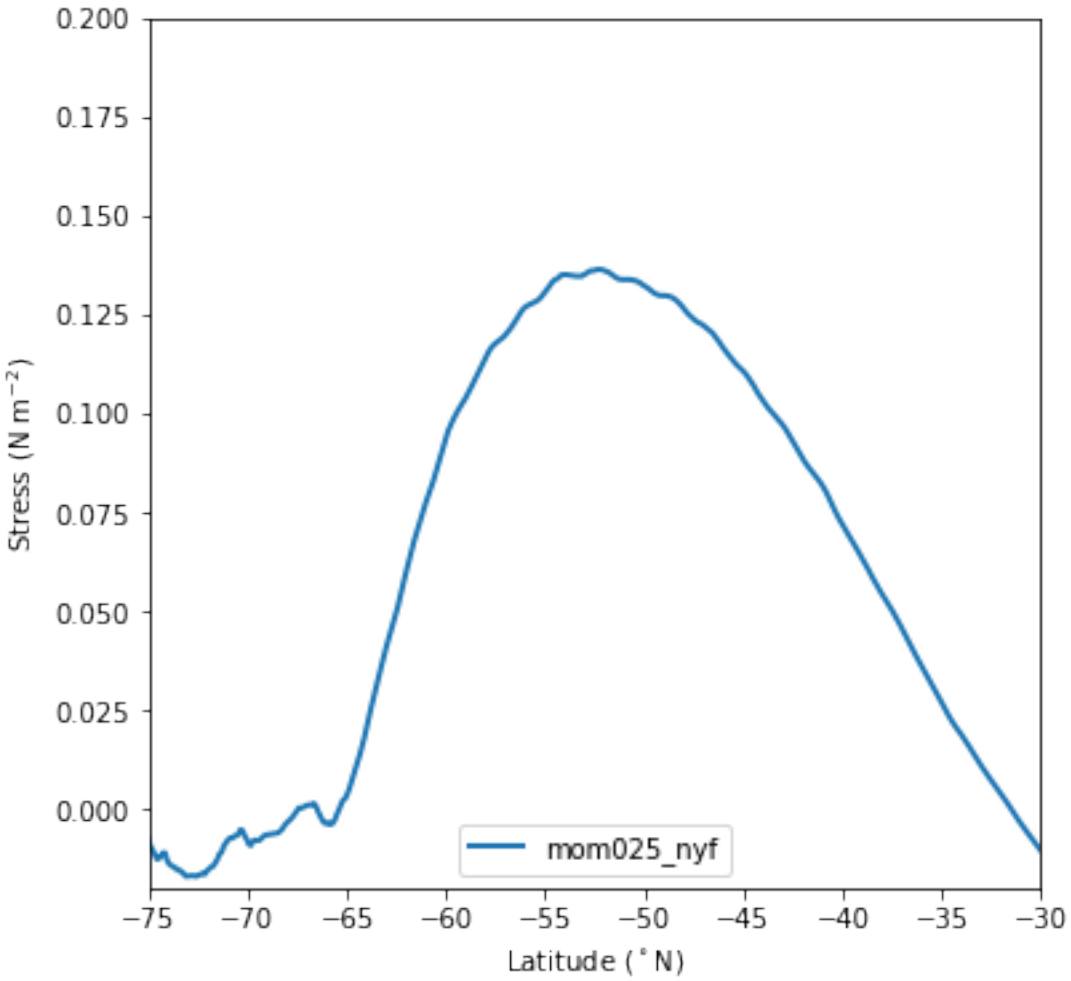
```
    plt.figure(figsize=(6,6))
for e in expts:
    ExpDir = os.path.join(DataDir,e)
    FileList = glob(os.path.join(ExpDir, 'output*/ocean_month.nc'))
    FileList.sort()

    dsx = xr.open_mfdataset(FileList, concat_dim='time',decode_times=False, engine=
↪'netcdf4')

    dsx.tau_x.mean('time').mean('xu_ocean').plot(linewidth=2,label=e)


plt.xlim([-75,-30])
plt.ylim([-0.02,0.2])
plt.xlabel('Latitude ($^\circ$N)')
plt.ylabel('Stress (N m$^{-2}$)')
```

```
plt.legend(loc=8,fontsize=10)
plt.savefig('figures/WindStress.pdf')
```

```
[#####################################  ] | 95% Completed |  7.6s
```

```
/g/data3/hh5/public/apps/miniconda3/envs/analysis3/lib/python3.5/site-packages/dask/
↪array/numpy_compat.py:45: RuntimeWarning: invalid value encountered in true_divide
  x = np.divide(x1, x2, out)
```

```
[########################################] | 100% Completed |  7.7s
CPU times: user 4.57 s, sys: 3.02 s, total: 7.58 s
Wall time: 8.86 s
```



Major Transports

In this section, we catalogue and save transport diagnostics from each of the major straits.

```
# location of Drake Passageand Indonesaian straits.
# Note that these are approximate -- need careful checking!!
```

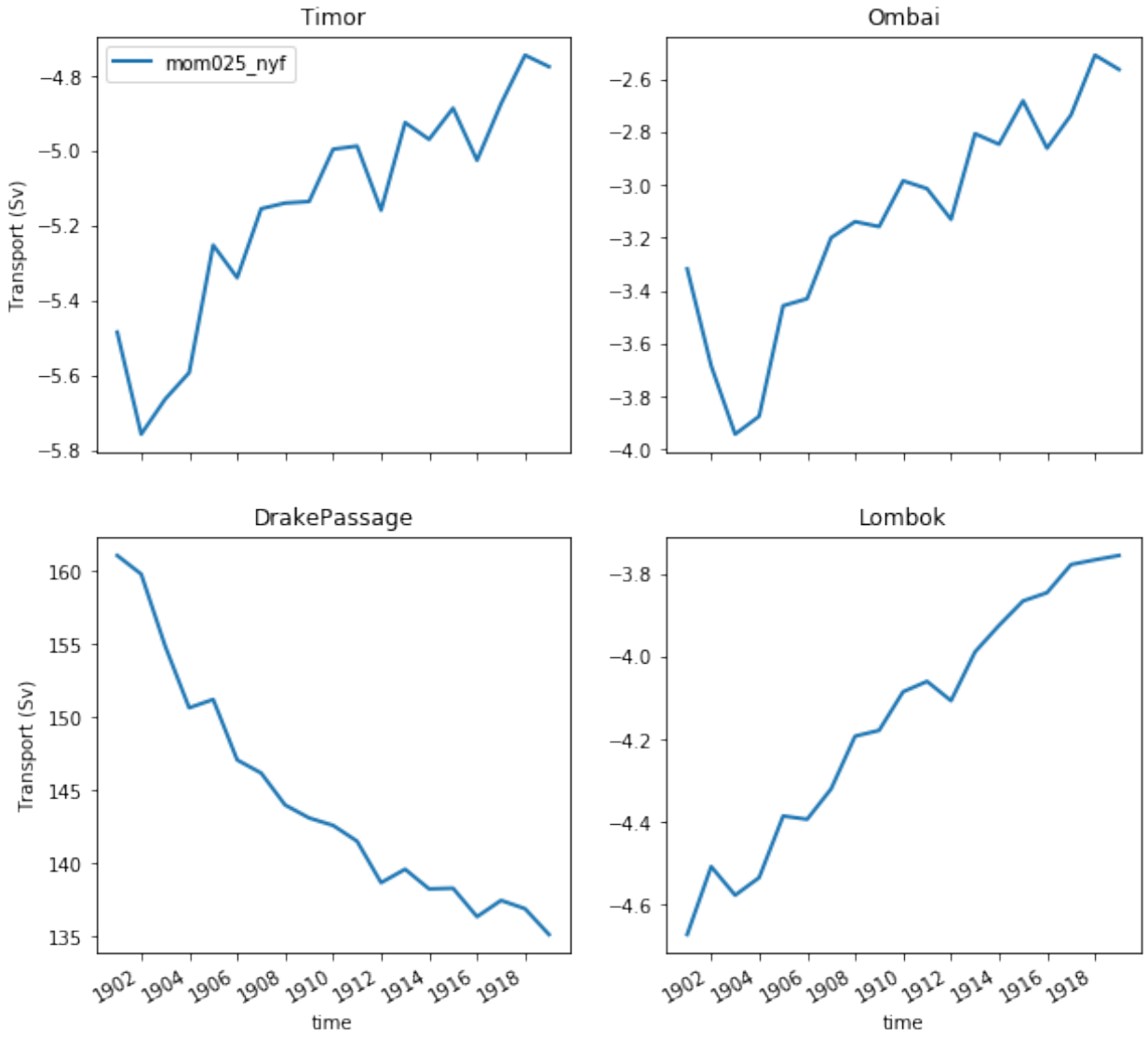
```
StraightDicts = { 'DrakePassage' :  {'xloc':840,'ymin':110,'ymax':250},
                  'Lombok' : {'yloc':463,'xmin':140,'xmax':146},
                  'Ombai' : {'xloc':174,'ymin':459,'ymax':465},
                  'Timor' : {'xloc':176,'ymin':446,'ymax':460},
                }
```

```
plt.figure(figsize=(10,10))
for expt in expts:
    # Find list of output directories
    ExpDir = os.path.join(DataDir, expt)
    FileList = glob(os.path.join(ExpDir, 'output*/ocean_month.nc'))
    FileList.sort()

    dsx = xr.open_mfdataset(FileList, decode_times=False, engine='netcdf4')
    dsx["time"].attrs["units"] = 'days since 1900-01-01'
    dsx = xr.decode_cf(dsx, decode_times=True)
    nplot = 0
    for straight in StraightDicts:
        nplot+=1
        print(nplot)

        TransDict = StraightDicts[straight]
        if 'xloc' in TransDict:
            tx_trans = dsx.tx_trans_int_z[:,TransDict['ymin']:TransDict['ymax'],
↪TransDict['xloc']]
            transport = tx_trans.sum('yt_ocean')
            valrange = tx_trans.valid_range[1]
        elif 'yloc' in TransDict:
            ty_trans = dsx.ty_trans_int_z[:,TransDict['yloc'],TransDict['xmin
↪']:TransDict['xmax']]
            transport = ty_trans.sum('xt_ocean')
            valrange = ty_trans.valid_range[1]
        else:
            transport = np.nan


        plt.subplot(2,2,nplot)
        transport.resample('a',dim='time').plot(label=expt, linewidth=2)
        #transport.plot(label=expt, linewidth=2)

        plt.title(straight)

plt.subplot(221)
plt.legend(loc='upper left')
plt.ylabel('Transport (Sv)')
plt.xlabel('')
plt.subplot(222)
plt.ylabel('')
plt.xlabel('')
plt.subplot(223)
plt.ylabel('Transport (Sv)')
plt.subplot(224)
plt.ylabel('')


plt.savefig('figures/Transports.pdf')
```

```
[########################################] | 100% Completed |  0.1s
[########################################] | 100% Completed |  0.1s
```

```
/g/data3/hh5/public/apps/miniconda3/envs/analysis3/lib/python3.5/site-packages/xarray/
↪conventions.py:389: RuntimeWarning: Unable to decode time axis into full numpy.
↪datetime64 objects, continuing using dummy netCDF4.datetime objects instead,␣
↪reason: dates out of range
  result = decode_cf_datetime(example_value, units, calendar)
```

```
[########################################] | 100% Completed |  0.1s
[########################################] | 100% Completed |  0.1s
1
[########################################] | 100% Completed |  3.0s
2
[########################################] | 100% Completed |  3.0s
3
[########################################] | 100% Completed |  3.0s
4
[########################################] | 100% Completed |  3.3s
```



Scalar Quantities

Look at some quantities that are saved in ocean_scalar.nc.

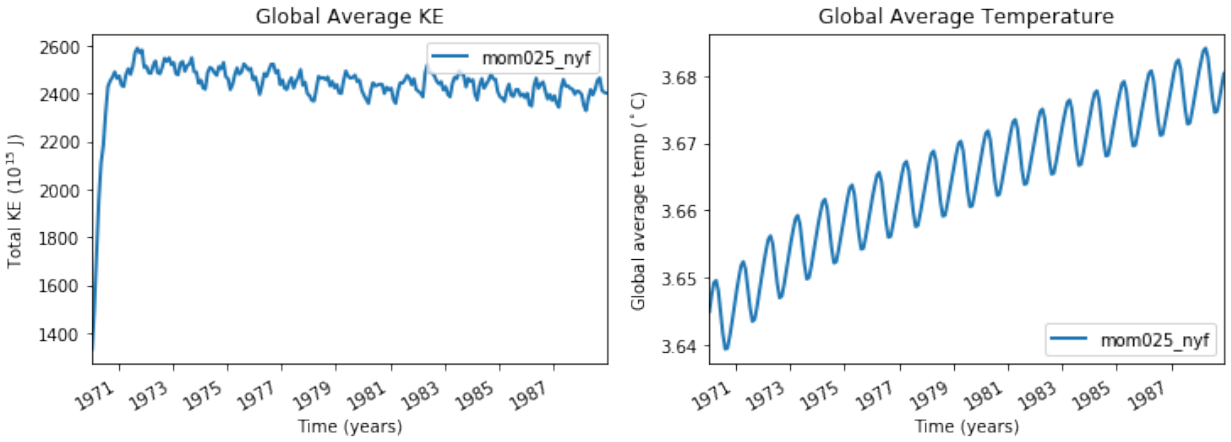
```
def extract_data(f):
    dataset = Dataset(f)

    data = {}
    for v in dataset.variables.values():
        if 'time' in v.dimensions and 'scalar_axis' in v.dimensions:
            data[v.name] = v[:].flatten()
    index = pd.to_datetime(dataset['time'][:], unit='D')
    df = pd.DataFrame(index=index, data=data)
    return df

scalar_quantities = {}
```

```
for expt in expts:
    expdir = os.path.join(DataDir,expt)
    DataFileList = glob(os.path.join(expdir, 'output*/ocean_scalar.nc'))

    frames = [ delayed(extract_data)(f) for f in DataFileList]
    if len(frames) > 0:
        scalar_quantities[expt] = delayed(pd.concat)(frames).compute(num_workers=4)
```

```
[########################################] | 100% Completed |  0.4s
```

```
plt.figure(figsize=(12,4))

for expt in expts:
    plt.subplot(121)
    scalar_quantities[expt].ke_tot.plot(label=expt, linewidth=2)

    plt.subplot(122)
    scalar_quantities[expt].temp_global_ave.plot(label=expt, linewidth=2)

plt.subplot(121)
plt.legend(loc='upper right')
plt.ylabel('Total KE (10$^{15}$ J)')
plt.xlabel('Time (years)')
plt.title('Global Average KE')

plt.subplot(122)
plt.legend(loc='lower right')
plt.ylabel('Global average temp ($^\circ$C)')
plt.xlabel('Time (years)')
plt.title('Global Average Temperature')

plt.savefig('figures/Scalars.pdf')
```

```
CPU times: user 591 ms, sys: 969 ms, total: 1.56 s
Wall time: 509 ms
```



Overturning Circulation

Next, let's look at overturning circulation in density space using ty_trans_rho. We will zonally average this diagnostic, without accounting for the tripolar grid, so ignore the Arctic.

```
def calculate_Ψ(expt, count = 4):
    ExpDir = os.path.join(DataDir, expt)

    FileList = glob(os.path.join(ExpDir, 'output*/ocean.nc'))
    FileList.sort()

    Ψ = 0
    for OceanFile in FileList[-count:]:
        ## I am having trouble with chunking, so have opted for the slower non-
↪chunking option until
        ## we can find a fix. -- AH 8/4/17.
        dsx = xr.open_dataset(OceanFile, decode_times=False, engine='netcdf4')
        #dsx = xr.open_dataset(OceanFile, decode_times=False, engine='netcdf4',␣
↪chunks={'potrho':5})

        psi_partial = dsx.ty_trans_rho.isel(time=0).sum('grid_xt_ocean').cumsum(
↪'potrho')
        dsx.close()

        Ψ = Ψ + psi_partial

    Ψ_avg = Ψ/count

    return Ψ_avg
```

```
def plot_psi(psi, expt, clev=np.arange(-20,20,2)):
    #x_formatter = matplotlib.ticker.ScalarFormatter(useOffset=False)

    plt.contourf(psi.grid_yu_ocean, psi.potrho, psi, cmap=plt.cm.PiYG,levels=clev,
↪extend='both')
    cb=plt.colorbar(orientation='vertical', shrink = 0.7)
    cb.ax.set_xlabel('Sv')
    plt.contour(psi.grid_yu_ocean, psi.potrho, psi, levels=clev, colors='k',␣
↪linewidths=0.25)
    plt.contour(psi.grid_yu_ocean, psi.potrho, psi, levels=[0.0,], colors='k',␣
↪linewidths=0.5)
    plt.gca().invert_yaxis()

    #plt.gca().yaxis.set_major_formatter(x_formatter)
    plt.ylim((1037.5,1034))
    plt.ylabel('Potential Density (kg m$^{-3}$)')
    plt.xlabel('Latitude ($^\circ$N)')
    plt.xlim([-75,85])
    plt.title('Overturning in %s' % expt)
```

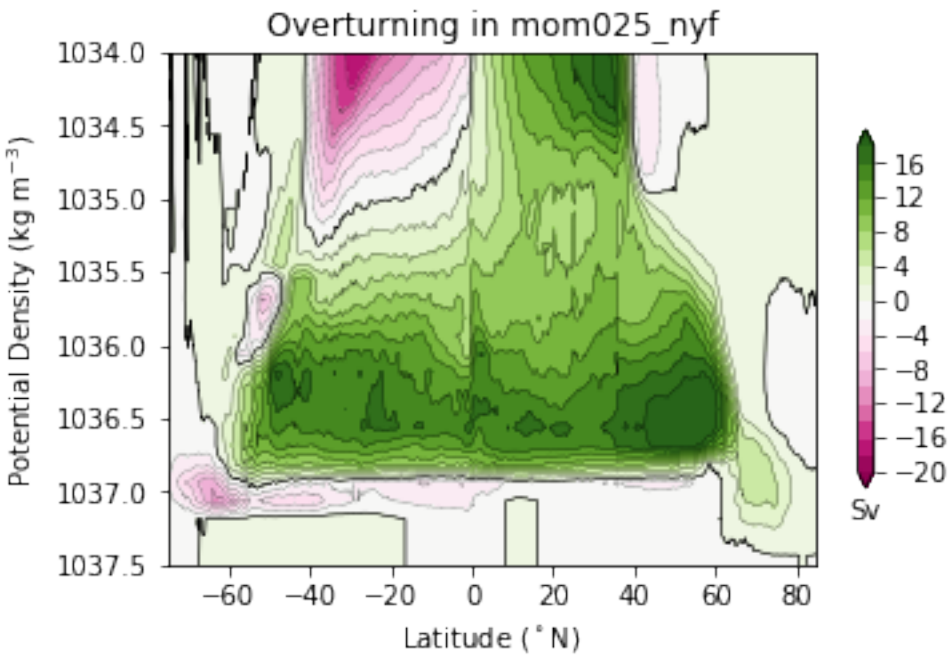
```
    ## One of our problems here is that the zonal sum and density-wise cumulative sum␣
↪in calculate_Ψ is so slow.
## A possible work-around is to calculate them once, save as a netCDF file, and then␣
↪simply read in multiple
## netcdf files when it is time to average them, or to construct timeseries.

## Here is a first cut of doing that ...
```

```
for expt in expts:
    # Find list of output directories
    ExpDir = os.path.join(DataDir, expt)
    FileList = glob(os.path.join(ExpDir, 'output*'))
    FileList.sort()

    for fn in FileList:
        MOCFile = os.path.join(fn, 'overturning.nc')

        # Have I Processed this one yet?
        if not os.path.exists(MOCFile):
            OceanFile = os.path.join(fn, 'ocean.nc')
            if os.path.exists(OceanFile):
                dsx = xr.open_dataset(OceanFile, decode_times=False, engine='netcdf4')
                psi = dsx.ty_trans_rho.isel(time=0).sum('grid_xt_ocean').cumsum(
↪'potrho')
                dsx.close()
                psi.to_netcdf(MOCFile, mode='w', engine='netcdf4')
                print('Saved ', MOCFile)
```

```
Saved  /g/data3/hh5/tmp/cosima/mom025/mom025_nyf/output018/overturning.nc
CPU times: user 4.53 s, sys: 361 ms, total: 4.89 s
Wall time: 5.54 s
```

```
plt.figure(figsize=(12,12))
nplot = 0
for expt in expts:
    nplot += 1
    ExpDir = os.path.join(DataDir, expt)
    FileList = glob(os.path.join(ExpDir, 'output*/overturning.nc'))
    FileList.sort()

    plt.subplot(3,2,nplot)
    dsx = xr.open_mfdataset(FileList, concat_dim='time',decode_times=False, engine=
↪'netcdf4')
    plot_psi(dsx.ty_trans_rho[-10:,:,:].mean('time'), expt)
```

```
[########################################] | 100% Completed |  0.2s
[########################################] | 100% Completed |  0.2s
[########################################] | 100% Completed |  0.2s
```




```
# plot timeseries of overturning
plt.figure(figsize=(12,10))
for expt in expts:
    ExpDir = os.path.join(DataDir, expt)
    FileList = glob(os.path.join(ExpDir, 'output*/overturning.nc'))
    FileList.sort()

    dsx = xr.open_mfdataset(FileList, concat_dim='time',decode_times=False, engine=
↪'netcdf4')
    dsx.time.attrs["units"] = 'days since 1900-01-01'
    dsx = xr.decode_cf(dsx, decode_times=True)

    plt.subplot(121)
    dsx.ty_trans_rho[:,60:,200].min('potrho').resample('A',dim='time').
↪plot(label=expt,linewidth=2)
    ## still need to check sensitivity to the exaction location in the line above

    #Southern branch of the AMOC at 35S
    plt.subplot(222)
    dsx.ty_trans_rho[:,60:,345].max('potrho').resample('A',dim='time').
↪plot(label=expt,linewidth=2)

    # AMOC at 26N
    plt.subplot(224)
    dsx.ty_trans_rho[:,60:,600].max('potrho').resample('A',dim='time').
↪plot(label=expt,linewidth=2)

#plt.title('AABW cell at 60S')
plt.subplot(121)
plt.legend(loc='lower left')
```

```
[########################################] | 100% Completed |  0.1s
[########################################] | 100% Completed |  0.1s
[########################################] | 100% Completed |  0.1s
```

```
<matplotlib.legend.Legend at 0x7f602d725fd0>
```

MOM-SIS 0.1° Diagnostics

This notebook calculates and retains key diagnostics from our mom01v5 simulations. The following experiments are included:

Experiment Name	Description
GFDL50	Original simulation with 50 vertical levels. Ran from Levitus for about 60 years, but data output only saved from about year 40.
KDS75	Branched from GFDL50 at year 45 (re-zeroed), but with Kial Stewart's 75 level scheme. Has now run for 103 years. Years 90-100 have 5-daily output.
KDS75_wind	Short (5-year) Antarctic wind perturbation case, branched from KDS75 at year 40.
KDS75_PI	Paul Spence's Poleward Intensification wind experiment. Branched from KDS75 at year 70, will run until year 100 with 5-daily output for the last decade
KDS75_UP	Paul Spence's Increased winds case. Branched from KDS75 at year 70, will run until year 100 with 5-daily output for the last decade. (In Progress)

Last updated March 30 2017.

https://github.com/OceansAus/cosima-cookbook.git

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/xarray/core/
↪formatting.py:16: FutureWarning: The pandas.tslib module is deprecated and will be␣
↪removed in a future version.
  from pandas.tslib import OutOfBoundsDatetime
```

```
import numpy as np
import matplotlib.pyplot as plt
import os

import dask
from dask import delayed
import dask.bag as db
import dask.array as da
import dask.dataframe as dd
from dask.diagnostics import ProgressBar

import netCDF4

import pandas as pd
import xarray as xr
from glob import glob
from mpl_toolkits.basemap import Basemap, shiftgrid

from IPython.display import display
from distributed.diagnostics import progress
import tempfile

from tqdm import tqdm_notebook
import tqdm
#import datashader
```

```
/short/v45/jm0634/conda/envs/cosima/lib/python3.6/site-packages/xarray/core/
↪formatting.py:16: FutureWarning: The pandas.tslib module is deprecated and will be␣
↪removed in a future version.
  from pandas.tslib import OutOfBoundsDatetime
```

```
DataDir = '/g/data3/hh5/tmp/cosima/mom01v5'
```

Experiments

```
expts = ['GFDL50','KDS75','KDS75_UP','KDS75_PI','KDS75_wind']
```

```
cachedir = tempfile.gettempdir()
```

```
from joblib import Memory
memory = Memory(cachedir=cachedir, verbose=0)
```

```
from dask.distributed import Client, LocalCluster
```

```
client = Client()
```

```
import warnings
warnings.filterwarnings("ignore",
                        message="Unable to decode time axis into full numpy.
↪datetime64 objects")
```

An experiment is a collection of outputNNN directories. Each directory represents the output of a single job submission script. These directories are created by the *payu* tool.

An experiment is given by a fully qualified path.

The file `diag_table` identifies which fields should be in the output directory.

But we can also examine the .nc files directly to infer their contents.

for each .nc file, get variables -> dimensions

.ncfile, varname, dimensions, chunksize

Generate an index for all netCDF4 files. The results are cached, so needs only to be done once.

```
@memory.cache
def build_index(expt):

    ExpDir = os.path.join(DataDir, expt)
    ncfiles = glob(os.path.join(ExpDir, 'output*/*.nc'))
    ncfiles.sort()

    def get_vars(ncpath):

        index = []

        ds = netCDF4.Dataset(ncpath)
        ncfile = os.path.basename(ncpath)

        for k, v in ds.variables.items():
            index.append( (v.name, v.dimensions,
                           tuple(v.chunking()), ncfile,
                           ncpath) )
        return index

    b = db.from_sequence(ncfiles)
    index = b.map(get_vars).concat()
    index = list(index)
    index = pd.DataFrame.from_records(index,
                                      columns = ['variable', 'dimensions',
                                                 'chunking', 'ncfile',
                                                   'path'])
    return index
```

Pregenerate index files for all output directories.

```
for expt in expts:
    build_index(expt)
```

```
def get_nc_variable(expt, ncfile, variable, chunks={}, n=None,
                    op= lambda x: x):
    """
```

```
    For a given experiment, concatenate together variable over all time
    given a basename ncfile.

    By default, xarray is set to use the same chunking pattern that is
    stored in the ncfile. This can be overwritten by passing in a dictionary
    chunks or setting chunks=None for no chunking (load directly into memory).

    n > 0 means only use the last n ncfiles files. Useful for testing.

    op() is function to apply to each variable before concatenating.

    """

    df = build_index(expt)
    var = df[(df.ncfile.str.contains(ncfile)) & (df.variable == variable)]

    chunking = var.chunking.iloc[0]
    dimensions = var.dimensions.iloc[0]
    default_chunks = dict(zip(dimensions, chunking))

    if chunks is not None:
        default_chunks.update(chunks)
        chunks = default_chunks

    ncfiles = sorted(list(var.path))

    if n is not None:
        ncfiles = ncfiles[-n:]

    b = db.from_sequence(ncfiles)
    b = b.map(lambda fn : op(xr.open_dataset(fn, chunks=chunks)[variable]) )
    datasets = b.compute()

    dsx = xr.concat(datasets, dim='time', coords='all')

    return dsx
```

Wind Stress fields

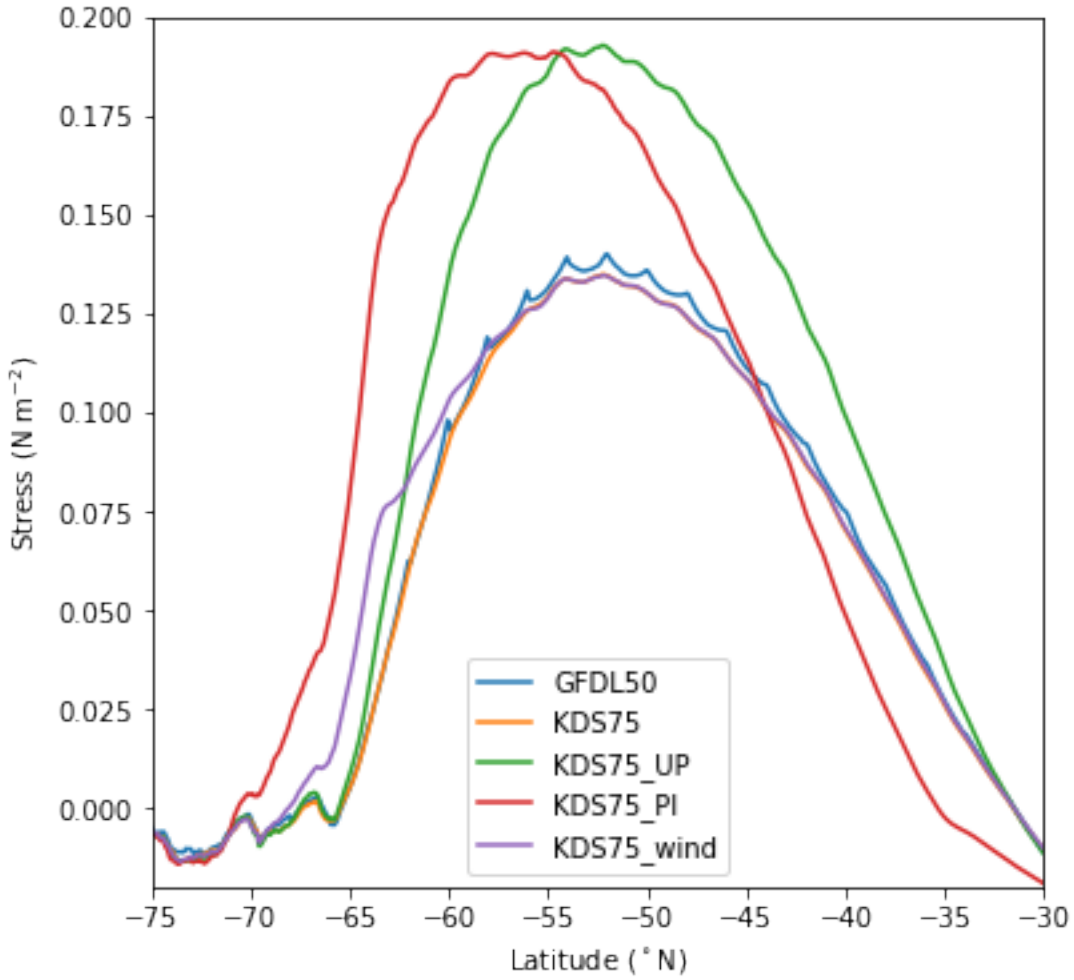
The following code block shows the zonal- and time-averaged wind stress forcing for each experiment.

```
@memory.cache
def calc_mean_tau_x(expt):

    tau_x = get_nc_variable(expt, 'ocean_month.nc', 'tau_x', n=25,
                            chunks={'xu_ocean':None})

    mean_tau_x = tau_x.mean('xu_ocean').mean('time')
    mean_tau_x = mean_tau_x.compute()
    mean_tau_x.name = e

    return mean_tau_x

plt.figure(figsize=(6,6))

for e in expts:
    print(e)
```

```
    mean_tau_x = calc_mean_tau_x(e)
    mean_tau_x.plot()

plt.xlim([-75,-30])
plt.ylim([-0.02,0.2])
plt.xlabel('Latitude ($^\circ$N)')
plt.ylabel('Stress (N m$^{-2}$)')
plt.legend(loc=8, fontsize=10)
```

```
GFDL50
KDS75
KDS75_UP
KDS75_PI
KDS75_wind
```

```
<matplotlib.legend.Legend at 0x7f43f0954978>
```



Major Transports

In this section, we catalogue and save transport diagnostics from each of the major straits.

Still to think about: * Should we put the background functions into a separate python file and load as a module?

```
straights = [ {'name': 'DrakePassage', 'xloc':2100,'ymin':225,'ymax':650},
              {'name': 'Lombok', 'yloc':1158,'xmin':354,'xmax':361},
              {'name': 'Ombai', 'xloc':449,'ymin':1152,'ymax':1163},
              {'name': 'Timor', 'xloc':440,'ymin':1125,'ymax':1145},
              {'name': 'Bering', 'yloc':2125,'xmin':1080,'xmax':1130},
              {'name': 'Denmark', 'yloc':2125,'xmin':2380,'xmax':2580},
            ]
```

```
@memory.cache
def calc_transport(expt, straight):
    ## Function to calculate barotropic transport across a given line of latitude or␣
↪longitude.
    ## Primarily designed for flow through straits.

    print('Calculating {}:{} transport'.format(expt, straight['name']))

    if 'xloc' in straight:

        ymin, ymax = straight['ymin'], straight['ymax']
        xloc = straight['xloc']

        op = lambda p: p.isel(xu_ocean=xloc)\
                        .isel(yt_ocean=slice(ymin, ymax))\
                        .chunk({'time':1})\
                        .sum('st_ocean').sum('yt_ocean')

        transport = get_nc_variable(expt, 'ocean.nc', 'tx_trans',
                                    chunks=None,
                                    op=op)

    elif 'yloc' in straight:

        xmin, xmax = straight['xmin'], straight['xmax']
        yloc = straight['yloc']

        op = lambda p: p.isel(yu_ocean=yloc)\
                        .isel(xt_ocean=slice(xmin, xmax))\
                        .chunk({'time':1})\
                        .sum('st_ocean').sum('xt_ocean')

        transport = get_nc_variable(expt, 'ocean.nc', 'ty_trans',
                                    chunks=None,
                                    op=op)
    else:
        print('Transports must be along lines of either constant latitude or longitude
↪')
        return None

    transport = transport.load()

    return transport
```

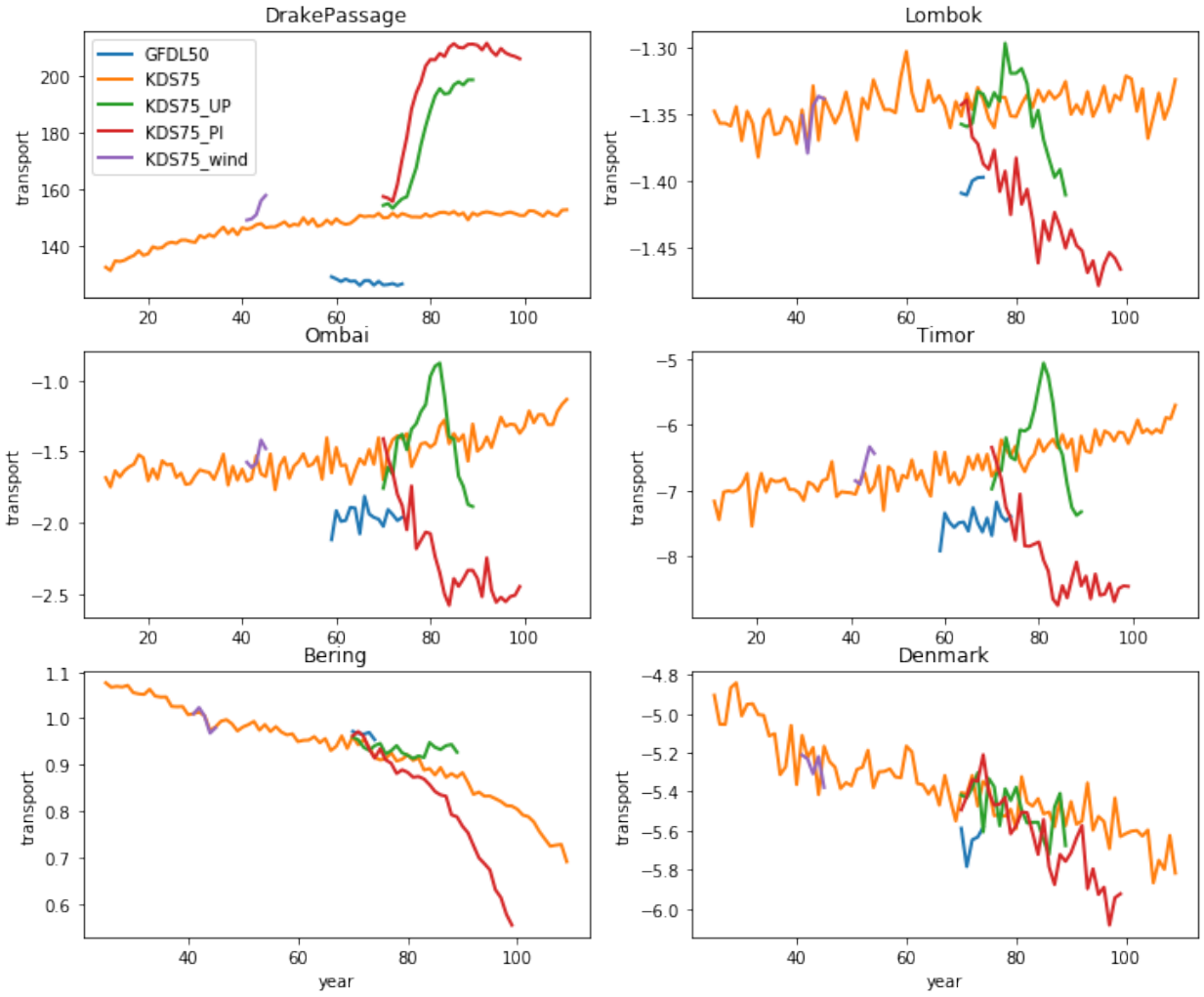
When distributing a computing with dask, you should try and minimize the number of task while keeping each task appropriate to a single work regarding memory usage.

```
plt.figure(figsize=(12,10))

for expt in expts:
    nplot = 0

    for straight in straights:
        nplot += 1

        transport = calc_transport(expt, straight)

        # see https://github.com/spencerahill/aospy/issues/98#issuecomment-256043833
        da = transport.to_dataset(name='transport')

        periods = []
        for date in da['time']:
            raw_date = date.values.item()
            periods.append(pd.Period(year=raw_date.year, month=raw_date.month,␣
↪day=raw_date.day, freq='D'))
        da['time'] = pd.PeriodIndex(periods)
        da = da.groupby('time.year').mean('time')
        transport = da.transport

        plt.subplot(3, 2, nplot)
        transport.plot(label=expt, linewidth=2)
        plt.title( straight['name'])

plt.subplot(321)
plt.legend(loc='upper left')
#plt.ylabel('Transport (Sv)')
plt.xlabel('')
plt.subplot(322)
#plt.ylabel('')
plt.xlabel('')
plt.subplot(323)
#plt.ylabel('Transport (Sv)')
plt.xlabel('')
plt.subplot(324)
#plt.ylabel('')
plt.xlabel('')
plt.subplot(325)
#plt.ylabel('Transport (Sv)')
#plt.xlabel('Time')
plt.subplot(326)
#plt.ylabel('')
#plt.xlabel('Time')
```

```
Calculating GFDL50:DrakePassage transport
Calculating GFDL50:Lombok transport
Calculating GFDL50:Ombai transport
Calculating GFDL50:Timor transport
Calculating GFDL50:Bering transport
Calculating GFDL50:Denmark transport
Calculating KDS75:DrakePassage transport
Calculating KDS75:Lombok transport
Calculating KDS75:Ombai transport
Calculating KDS75:Timor transport
Calculating KDS75:Bering transport
Calculating KDS75:Denmark transport
```

```
Calculating KDS75_UP:DrakePassage transport
Calculating KDS75_UP:Lombok transport
Calculating KDS75_UP:Ombai transport
Calculating KDS75_UP:Timor transport
Calculating KDS75_UP:Bering transport
Calculating KDS75_UP:Denmark transport
Calculating KDS75_PI:DrakePassage transport
Calculating KDS75_PI:Lombok transport
Calculating KDS75_PI:Ombai transport
Calculating KDS75_PI:Timor transport
Calculating KDS75_PI:Bering transport
Calculating KDS75_PI:Denmark transport
Calculating KDS75_wind:DrakePassage transport
Calculating KDS75_wind:Lombok transport
Calculating KDS75_wind:Ombai transport
Calculating KDS75_wind:Timor transport
Calculating KDS75_wind:Bering transport
Calculating KDS75_wind:Denmark transport
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f43f002a828>
```



Scalar Quantities

Look at some quantities that are saved in ocean_scalar.nc.

```
def extract_data(f):
    dataset = netCDF4.Dataset(f)

    data = {}
    for v in dataset.variables.values():
        if 'time' in v.dimensions and 'scalar_axis' in v.dimensions:
            data[v.name] = v[:].flatten()
    index = pd.to_datetime(dataset['time'][:], unit='D')
    df = pd.DataFrame(index=index, data=data)
    return df

@memory.cache
def calc_scalar_quantities(expt):
    print(expt)

    df = build_index(expt)

    # identify all scalar variables
    df = df[(df.ncfile == 'ocean_scalar.nc') & (df.dimensions == ('time', 'scalar_axis
↪' )) ]
    ncfiles = sorted(list(df.path))

    b = db.from_sequence(ncfiles)
    b = list(b.map(extract_data))
    df = pd.concat(b)

    return df

scalar_quantities = {}
for expt in expts:

    scalar_quantities[expt] = calc_scalar_quantities(expt)
```

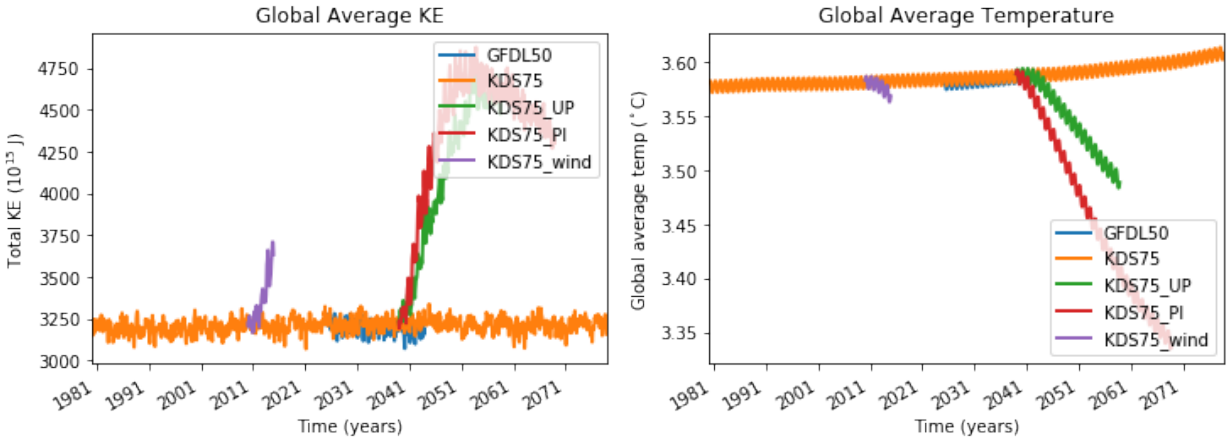
```
GFDL50
KDS75
KDS75_UP
KDS75_PI
KDS75_wind
```

```
scalar_quantities['GFDL50'].describe().T
```

```
plt.figure(figsize=(12,4))

for expt in expts:

    plt.subplot(121)
    scalar_quantities[expt].ke_tot.plot(label=expt, linewidth=2)

    plt.subplot(122)
    scalar_quantities[expt].temp_global_ave.plot(label=expt, linewidth=2)

plt.subplot(121)
plt.legend(loc='upper right')
```

```
plt.ylabel('Total KE (10$^{15}$ J)')
plt.xlabel('Time (years)')
plt.title('Global Average KE')

plt.subplot(122)
plt.legend(loc='lower right')
plt.ylabel('Global average temp ($^\circ$C)')
plt.xlabel('Time (years)')
plt.title('Global Average Temperature')
```

```
CPU times: user 2.37 s, sys: 34.9 ms, total: 2.4 s
Wall time: 2.57 s
```



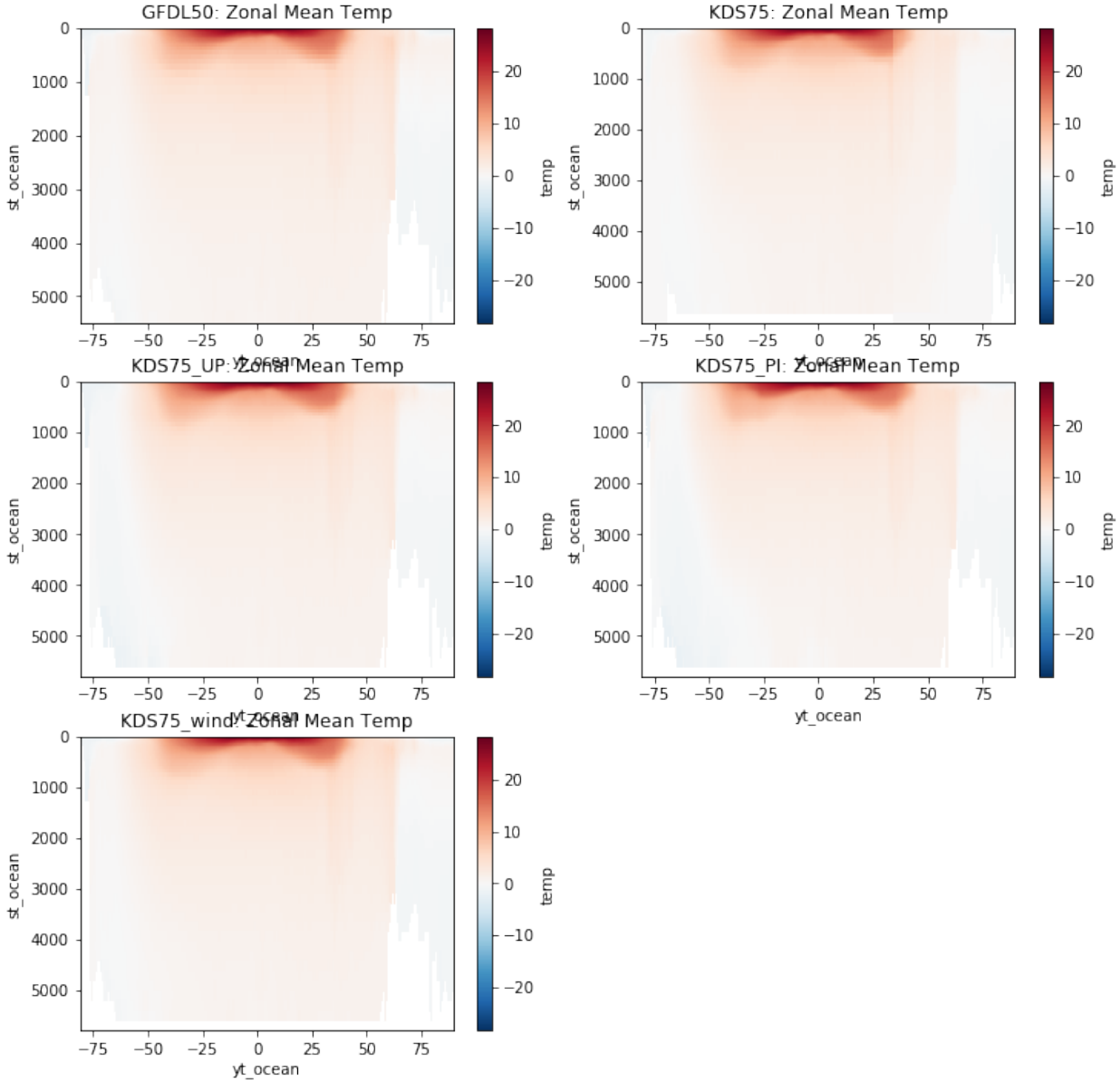

```
@memory.cache
def calc_zonal_mean_temp(expt):
    print('Calculating {} zonal_mean_temp'.format(expt))

    if expt == 'KDS75':
        ncfile = 'ocean_month.nc'
    else:
        ncfile = 'ocean.nc'

    zonal_temp = get_nc_variable(expt, ncfile, 'temp',
                                 chunks={'st_ocean': None},
                                 n=25)

    zonal_mean_temp = zonal_temp.mean('xt_ocean').mean('time')
    zonal_mean_temp.load()

    return zonal_mean_temp
```

```
def plot_zonal_mean_temp(expt):
    zonal_mean_temp = calc_zonal_mean_temp(expt)

    zonal_mean_temp.plot()
    plt.gca().invert_yaxis()
    plt.title('{}: Zonal Mean Temp'.format(expt))
```

```
plt.figure(figsize=(12,12))
nplot = 0
```

```
for expt in expts:

    nplot += 1
    plt.subplot(3,2,nplot)

    plot_zonal_mean_temp(expt)
```

```
CPU times: user 718 ms, sys: 66.3 ms, total: 785 ms
Wall time: 772 ms
```



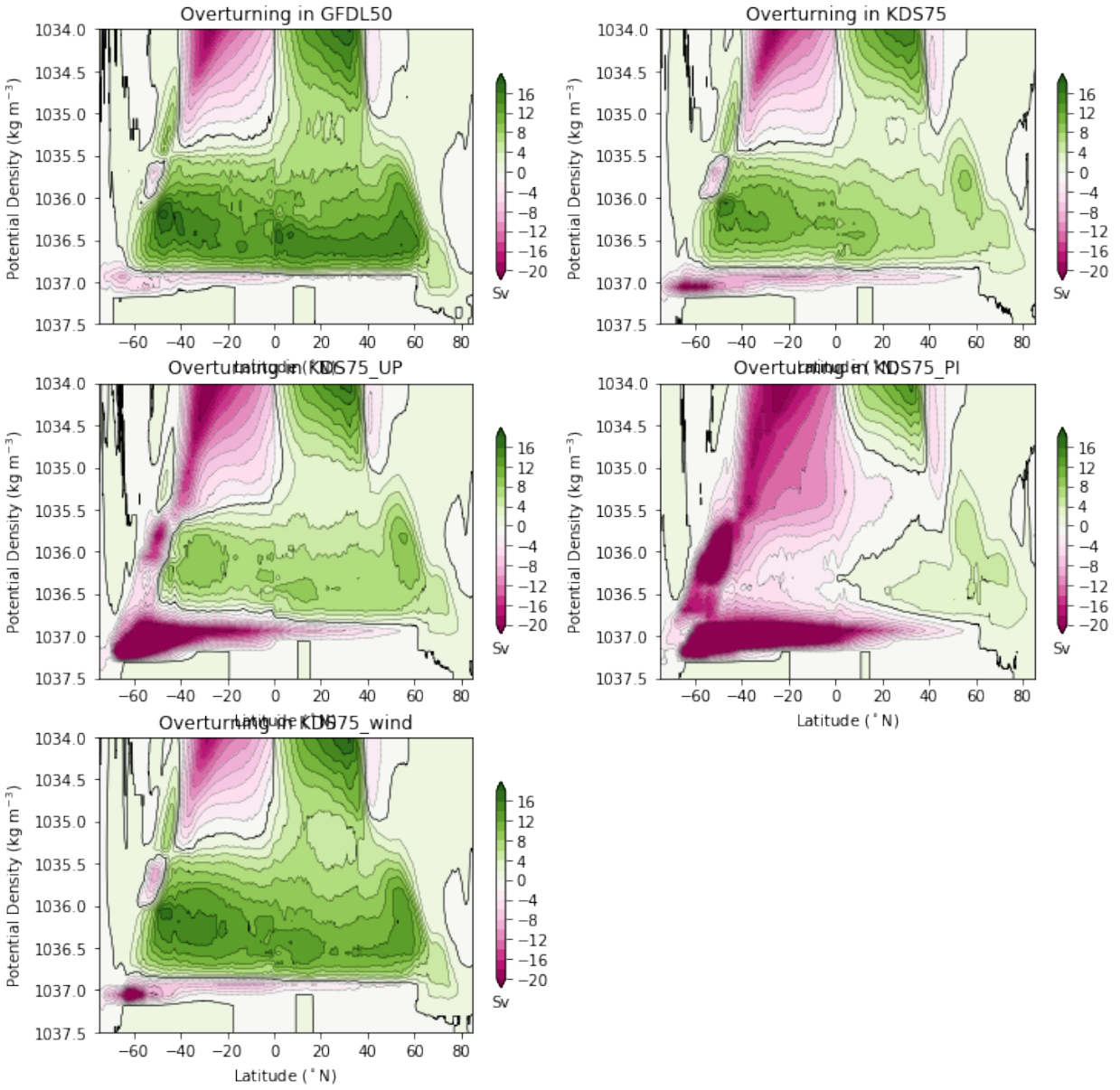
Overturning Circulation

Next, let's look at overturning circulation in density space using ty_trans_rho. We will zonally average this diagnostic, without accounting for the tripolar grid, so ignore the Arctic.

```
@memory.cache
def calc_psi_avg(expt):
    print('Calculating {} psi_avg'.format(expt))

    op = lambda p: p.sum('grid_xt_ocean').cumsum('potrho')

    psi = get_nc_variable(expt, 'ocean.nc', 'ty_trans_rho',
                          op=op,
                          chunks={'potrho': None}, n=25)

    psi_avg = psi.mean('time')
    psi_avg = psi_avg.compute()

    return psi_avg
```

```
def plot_psi(psi_avg, expt, clev=np.arange(-20,20,2)):

    plt.contourf(psi_avg.grid_yu_ocean,
                 psi_avg.potrho,
                 psi_avg,
                 cmap=plt.cm.PiYG,levels=clev,extend='both')
    cb=plt.colorbar(orientation='vertical', shrink = 0.7)
    cb.ax.set_xlabel('Sv')
    plt.contour(psi_avg.grid_yu_ocean,
                psi_avg.potrho,
                psi_avg, levels=clev, colors='k', linewidths=0.25)
    plt.contour(psi_avg.grid_yu_ocean,
                psi_avg.potrho, psi_avg,
                levels=[0.0,], colors='k', linewidths=0.5)
    plt.gca().invert_yaxis()

    plt.ylim((1037.5,1034))
    plt.ylabel('Potential Density (kg m$^{-3}$)')
    plt.xlabel('Latitude ($^\circ$N)')
    plt.xlim([-75,85])
    plt.title('Overturning in %s' % expt)
```

```
plt.figure(figsize=(12,12))
nplot = 0
for expt in expts:

    psi_avg = calc_psi_avg(expt)
    nplot += 1

    plt.subplot(3,2,nplot)
    plot_psi(psi_avg, expt)
```



Eddy Kinetic Energy

Plan here is to add: * maps of vertically summed EKE (actually, not sure this is even feasible for this dataset) * timeseries of globally averaged EKE

We can only do this for portions of simulations which have 5-day average velocities saved, which means directories with `ocean__*.nc` files.

```
@memory.cache
def calc_eke(expt, box_index):
    yi, xi = box_index

    box = {'yu_ocean': slice(300*yi, 300*(yi+1)),
           'xu_ocean': slice(400*xi, 400*(xi+1))}
```

```
    op = lambda p: p.isel(**box)

    u = get_nc_variable(expt, 'ocean__', 'u', op=op, n=72)
    v = get_nc_variable(expt, 'ocean__', 'v', op=op, n=72)

    u_avg = u.mean('time')
    v_avg = v.mean('time')

    MKE = 0.5 * (u_avg**2 + v_avg**2)
    MKE = MKE.sum(dim='st_ocean')
    MKE = MKE.to_dataset(name='MKE')

    u_ = u - u_avg
    v_ = v - v_avg

    EKE = 0.5 * (u_**2 + v_**2)

    EKE = EKE.sum(dim='st_ocean')
    EKE = EKE.to_dataset(name='EKE')

    dsx = xr.merge([MKE, EKE])
    dsx.load()

    return dsx

expt = 'KDS75'
calc_eke(expt, (0,0))
```

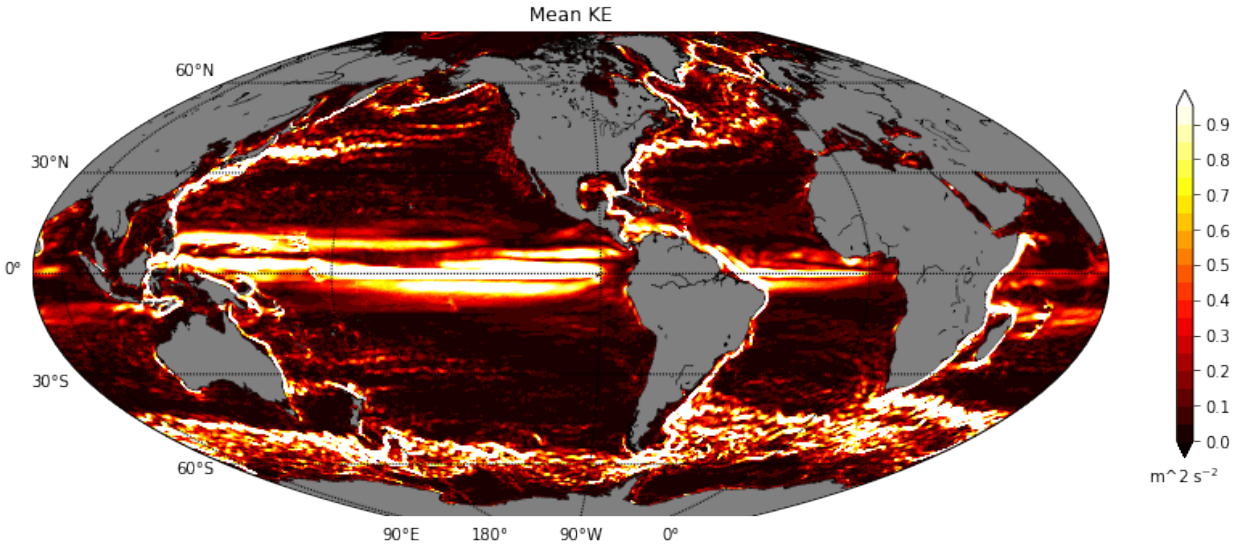
```
<xarray.Dataset>
Dimensions:   (time: 72, xu_ocean: 400, yu_ocean: 300)
Coordinates:
  * xu_ocean  (xu_ocean) float64 -279.9 -279.8 -279.7 -279.6 -279.5 -279.4 ...
  * yu_ocean  (yu_ocean) float64 -81.09 -81.05 -81.0 -80.96 -80.92 -80.88 ...
  * time      (time) object   99-01-03 12:00:00   99-01-08 12:00:00 ...
    geolon_c  (time, yu_ocean, xu_ocean) float64 nan nan nan nan nan nan nan ...
    geolat_c  (time, yu_ocean, xu_ocean) float64 nan nan nan nan nan nan nan ...
Data variables:
    MKE       (yu_ocean, xu_ocean) float64 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ...
    EKE       (time, yu_ocean, xu_ocean) float64 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ...
```

```
from itertools import product

# ceil(x/y) = (x+y+1)//y
yi = range(2700//(300))
xi = range(3600//(400))

box_indexes = list(product(*[yi, xi]))
```

```
# Plot in basemap

plt.figure(figsize=(15,6))
lev = np.arange(0, 1.0, 0.05)
map = Basemap(projection='mbtfpq',
              lon_0 = -100, resolution='l')
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
```

```
map.drawparallels(np.arange(-60.,61.,30.),
                  labels=[True,False,False,False])
map.drawmeridians(np.arange(-180.,181.,90.),
                  labels=[False,False,False,True])

for box_index in tqdm_notebook(box_indexes):
    #print(box_index)
    dsx = calc_eke(expt, box_index)

    x=dsx.xu_ocean[:]
    y=dsx.yu_ocean[:]
    lon, lat = np.meshgrid(x, y)

    X, Y = map(lon,lat)

    map.contourf(X, Y, dsx.MKE,
                 cmap=plt.cm.hot,
                 levels=lev,extend='both')

cb = plt.colorbar(orientation='vertical',shrink = 0.7)

cb.ax.set_xlabel('m^2 s$^{-2}$')
plt.title('Mean KE')
```

```
<matplotlib.text.Text at 0x7f4386801a90>
```

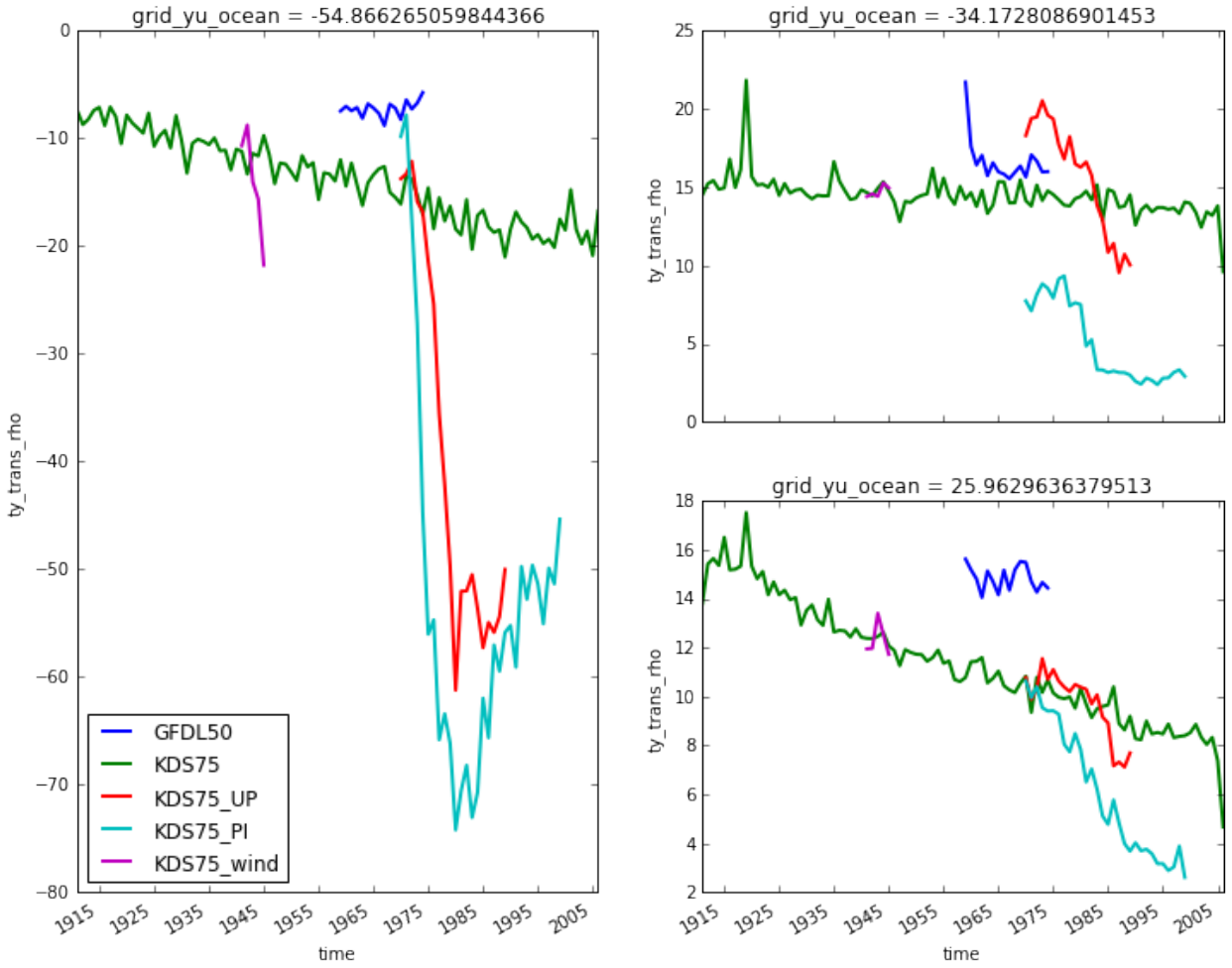



```
# for testing plotting, etc.
plt.figure(figsize=(12,10))
for expt in expts:
    ExpDir = os.path.join(DataDir, expt)
    FileList = glob(os.path.join(ExpDir, 'output*/overturning.nc'))
    FileList.sort()

    dsx = xr.open_mfdataset(FileList, concat_dim='time',decode_times=False, engine=
↪'netcdf4')
    dsx.time.attrs["units"] = 'days since 1900-01-01'
    dsx = xr.decode_cf(dsx, decode_times=True)
```

```
    plt.subplot(121)
    dsx.ty_trans_rho[:,60:,585].min('potrho').resample('A',dim='time').
↪plot(label=expt,linewidth=2)
    ## still need to check sensitivity to the exaction location in the line above
    print(dsx.potrho[60].values)

    #Southern branch of the AMOC at 35S
    plt.subplot(222)
    dsx.ty_trans_rho[:,60:,880].max('potrho').resample('A',dim='time').
↪plot(label=expt,linewidth=2)

    # AMOC at 26N
    plt.subplot(224)
    dsx.ty_trans_rho[:,60:,1513].max('potrho').resample('A',dim='time').
↪plot(label=expt,linewidth=2)

#plt.title('AABW cell at 60S')
plt.subplot(121)
plt.legend(loc='lower left')
```

```
[########################################] | 100% Completed |  0.2s
1035.5625
[########################################] | 100% Completed |  0.2s
[########################################] | 100% Completed |  0.3s
[########################################] | 100% Completed |  1.5s
1035.5625
[########################################] | 100% Completed |  1.3s
[########################################] | 100% Completed |  1.4s
[########################################] | 100% Completed |  0.3s
1035.5625
[########################################] | 100% Completed |  0.3s
[########################################] | 100% Completed |  0.4s
[########################################] | 100% Completed |  0.5s
1035.5625
[########################################] | 100% Completed |  0.5s
[########################################] | 100% Completed |  0.5s
[########################################] | 100% Completed |  0.1s
1035.5625
[########################################] | 100% Completed |  0.1s
[########################################] | 100% Completed |  0.1s
```

```
<matplotlib.legend.Legend at 0x7f131a5149b0>
```



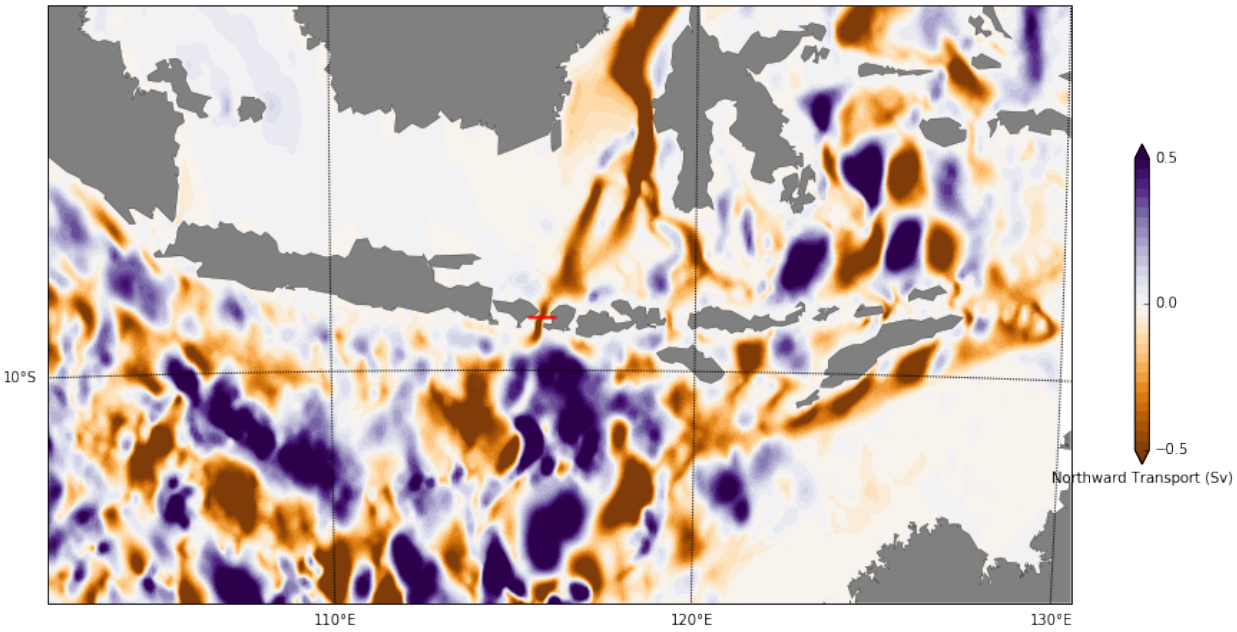
Indo Throughflow and other Straits

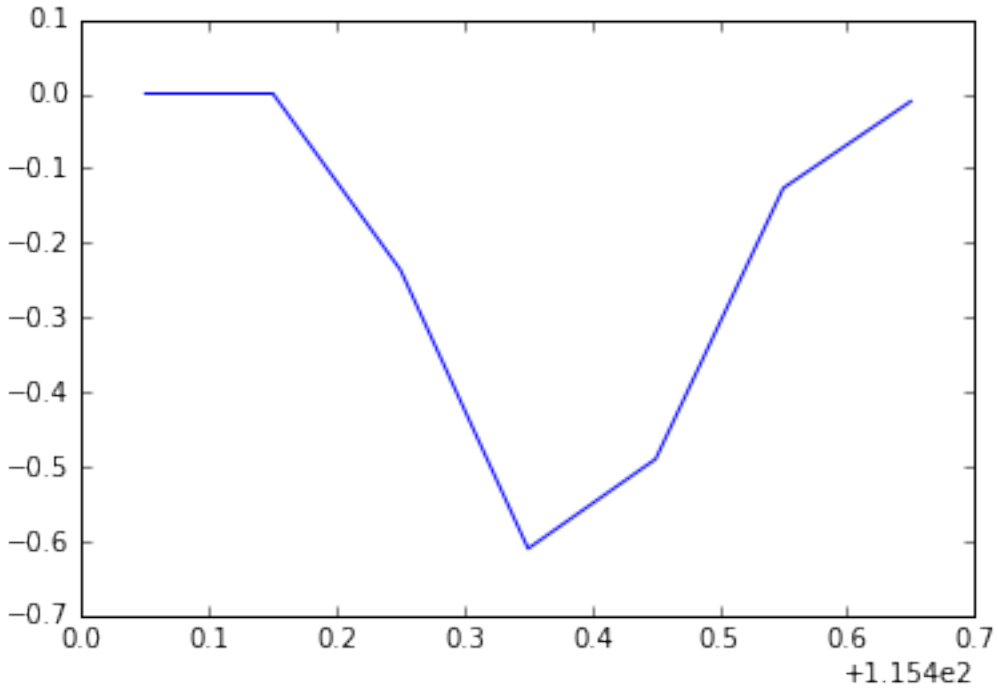
- THis is pretty much done now, but I'm retaining these files to document how we calculate transport for each strait.

```
## test code to play with computation of lombok transport

# load ty_trans from ocean.nc
OceanFile = os.path.join(DataDir,expts[1],'output383/ocean.nc')
dsx = xr.open_dataset(OceanFile, decode_times=False, engine='netcdf4')
ty_trans = dsx.ty_trans[0,:,:1500,:500]
x = dsx.xt_ocean[:500]
y = dsx.yu_ocean[:1500]

# vertically integrate
transport = ty_trans.sum(axis=0)

# plot local regions around Lombok, etc.
plt.figure(figsize=(16,8))
levels = np.linspace(-0.5,0.5,31)
map = Basemap(llcrnrlon=-258,llcrnrlat=-16,urcrnrlon=-230,urcrnrlat=0,projection=
↪'stere',resolution='l',lon_0=-245,lat_0=-7)
```

```
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
# draw parallels and meridians.
map.drawparallels(np.arange(-80.,81.,10.),labels=[True,False,False,True])
map.drawmeridians(np.arange(-180.,181.,10.),labels=[True,False,False,True])
xv, yv = np.meshgrid(x, y)
X, Y = map(xv,yv)
map.contourf(X,Y,transport, cmap=plt.cm.PuOr, levels=levels, extend='both') #CMRmap_r,
↪ gist_stern_r
cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-1, -0.5, 0, 0.5, 1 ])
cb.ax.set_xlabel('Northward Transport (Sv)')

# pick exact location and sum for each strait
LombokDict = {'yloc':1158,'xmin':354,'xmax':361}
xx = np.array([X[LombokDict['yloc'],LombokDict['xmin']], X[LombokDict['yloc'],
↪LombokDict['xmax']]])+360
yy = np.array([Y[LombokDict['yloc'],LombokDict['xmin']],Y[LombokDict['yloc'],
↪LombokDict['xmax']]])
#print(xx)
#print(yy)

map.plot(xx,yy,linewidth=2,color='r')


plt.figure()
plt.plot(x[354:361]+360,transport[LombokDict['yloc'],LombokDict['xmin']:LombokDict[
↪'xmax']])
lombok_transport = np.sum(transport[LombokDict['yloc'],LombokDict['xmin']:LombokDict[
↪'xmax']])

print(lombok_transport)
```

```
<xarray.DataArray 'ty_trans' ()>
array(-1.473966432557063)
Coordinates:
    time      float64 3.591e+04
    yu_ocean  float64 -8.568
```

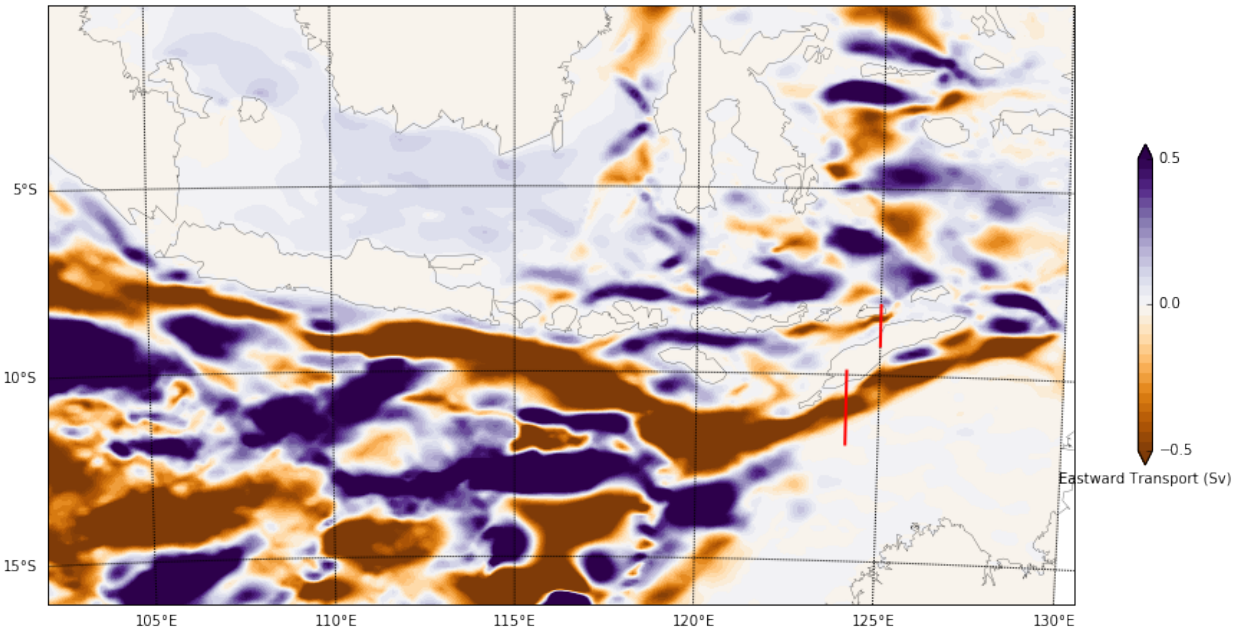


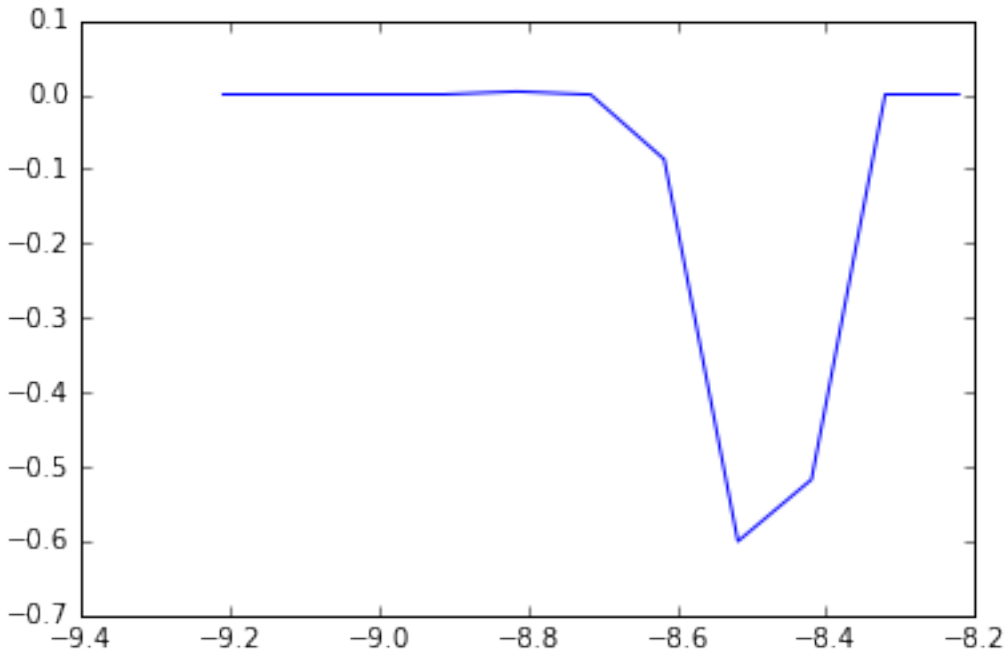


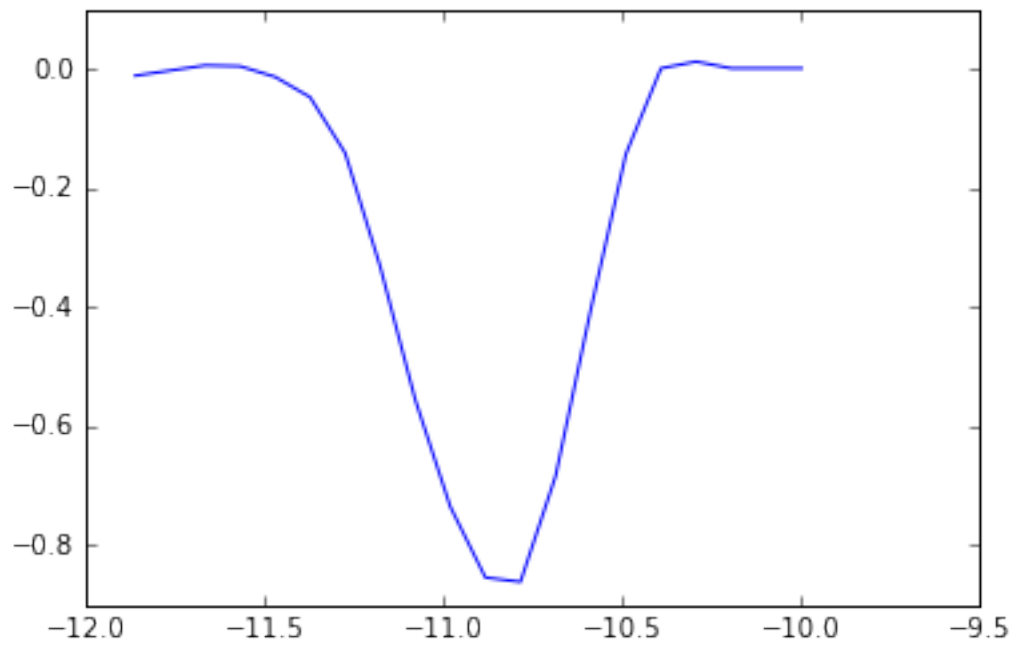

```
# load ty_trans from ocean.nc
OceanFile = os.path.join(DataDir,expts[1],'output385/ocean.nc')
dsx = xr.open_dataset(OceanFile, decode_times=False, engine='netcdf4')
tx_trans = dsx.tx_trans[0,:,:1500,:500]
x = dsx.xu_ocean[:500]
y = dsx.yt_ocean[:1500]
transport = tx_trans.sum(axis=0)
xv, yv = np.meshgrid(x, y)

plt.figure(figsize=(16,8))
```

```
levels = np.linspace(-0.5,0.5,31)
map = Basemap(llcrnrlon=-258,llcrnrlat=-16,urcrnrlon=-230,urcrnrlat=0,projection=
↪'stere',resolution='l',lon_0=-245,lat_0=-7)
map.drawcoastlines(linewidth=0.25)
#map.fillcontinents(color='gray',lake_color='gray')
# draw parallels and meridians.
map.drawparallels(np.arange(-80.,81.,5.),labels=[True,False,False,True])
map.drawmeridians(np.arange(-180.,181.,5.),labels=[True,False,False,True])
X, Y = map(xv,yv)
map.contourf(X,Y,transport, cmap=plt.cm.PuOr, levels=levels, extend='both') #CMRmap_r,
↪ gist_stern_r
cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-1, -0.5, 0, 0.5, 1 ])
cb.ax.set_xlabel('Eastward Transport (Sv)')


# pick exact location and sum for each strait
# This for Ombai
OmbaiDict = {'xloc':449,'ymin':1152,'ymax':1163}
xx = np.array([X[OmbaiDict['ymin'],OmbaiDict['xloc']], X[OmbaiDict['ymax'],OmbaiDict[
↪'xloc']]])+360
yy = np.array([Y[OmbaiDict['ymin'],OmbaiDict['xloc']],Y[OmbaiDict['ymax'],OmbaiDict[
↪'xloc']]])
map.plot(xx,yy,linewidth=2,color='r')

# This for Timor
TimorDict = {'xloc':440,'ymin':1125,'ymax':1145}
xx = np.array([X[TimorDict['ymin'],TimorDict['xloc']], X[TimorDict['ymax'],TimorDict[
↪'xloc']]])+360
yy = np.array([Y[TimorDict['ymin'],TimorDict['xloc']],Y[TimorDict['ymax'],TimorDict[
↪'xloc']]])
map.plot(xx,yy,linewidth=2,color='r')

# plot and measure ombai
plt.figure()
plt.plot(y[OmbaiDict['ymin']:OmbaiDict['ymax']],transport[OmbaiDict['ymin']:OmbaiDict[
↪'ymax'],OmbaiDict['xloc']])
ombai_transport = np.sum(transport[OmbaiDict['ymin']:OmbaiDict['ymax'],OmbaiDict['xloc
↪']])
print(ombai_transport)

# plot and measure timor
plt.figure()
plt.plot(y[TimorDict['ymin']:TimorDict['ymax']],transport[TimorDict['ymin']:TimorDict[
↪'ymax'],TimorDict['xloc']])
timor_transport = np.sum(transport[TimorDict['ymin']:TimorDict['ymax'],TimorDict['xloc
↪']])
print(timor_transport)
```

```
<xarray.DataArray 'tx_trans' ()>
array(-1.2015297539478524)
Coordinates:
    time      float64 3.609e+04
    xu_ocean  float64 -235.0
<xarray.DataArray 'tx_trans' ()>
array(-4.769188458296412)
Coordinates:
    time      float64 3.609e+04
    xu_ocean  float64 -235.9
```


5°S
10°S
15°S
105°E
110°E
115°E
120°E
125°E
130°E
0.5
0.0
−0.5
Eastward Transport (Sv)

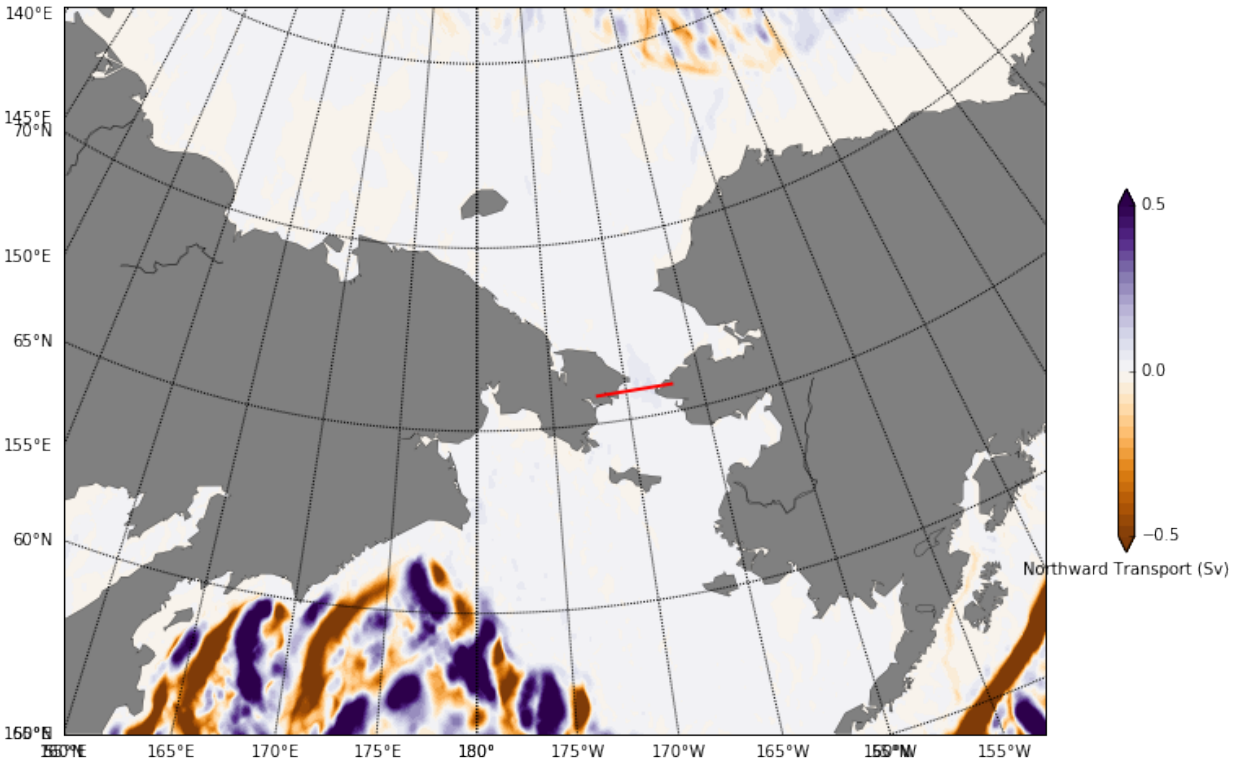

0.1
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−9.4
−9.2
−9.0
−8.8
−8.6
−8.4
−8.2

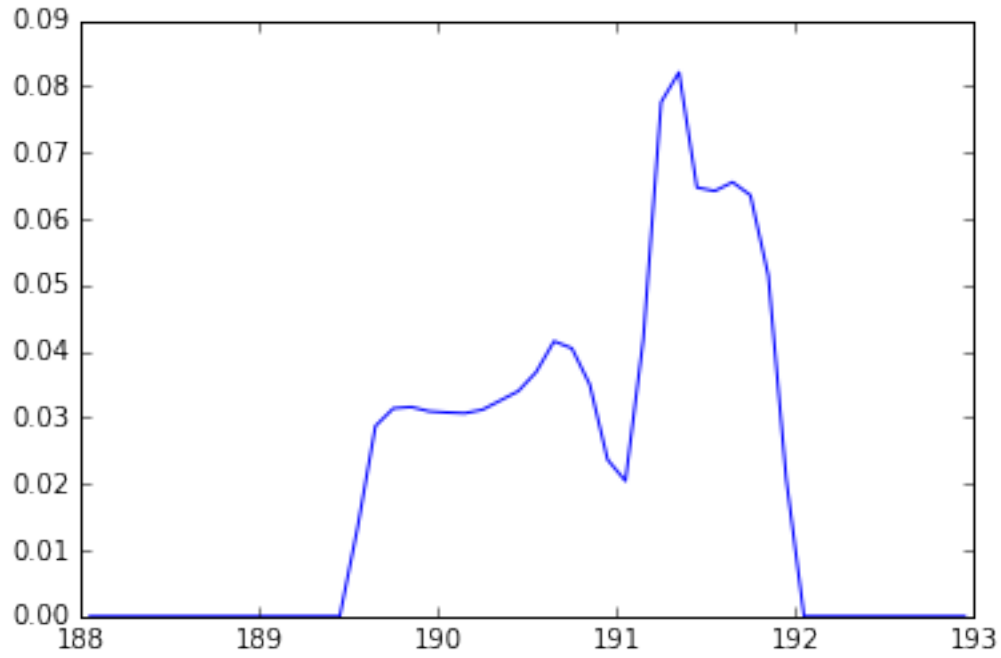



```
# Bering Strait
# load ty_trans from ocean.nc
OceanFile = os.path.join(DataDir,expts[1],'output383/ocean.nc')
dsx = xr.open_dataset(OceanFile, decode_times=False, engine='netcdf4')
ty_trans = dsx.ty_trans[0,:,:,:1500]
x = dsx.xt_ocean[:1500]
y = dsx.yu_ocean[:]

# vertically integrate
transport = ty_trans.sum(axis=0)

# plot local regions around Lombok, etc.
plt.figure(figsize=(16,8))
levels = np.linspace(-0.5,0.5,31)
map = Basemap(llcrnrlon=-200,llcrnrlat=55,urcrnrlon=-130,urcrnrlat=70,projection=
↪'stere',resolution='l',lon_0=-180,lat_0=60)
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
# draw parallels and meridians.
map.drawparallels(np.arange(-80.,81.,5.),labels=[True,False,False,True])
map.drawmeridians(np.arange(-180.,181.,5.),labels=[True,False,False,True])
xv, yv = np.meshgrid(x, y)
X, Y = map(xv,yv)
map.contourf(X,Y,transport, cmap=plt.cm.PuOr, levels=levels, extend='both') #CMRmap_r,
↪ gist_stern_r
cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-1, -0.5, 0, 0.5, 1 ])
cb.ax.set_xlabel('Northward Transport (Sv)')

# pick exact location and sum for each strait
BeringDict = {'yloc':2125,'xmin':1080,'xmax':1130}
xx = np.array([X[BeringDict['yloc'],BeringDict['xmin']], X[BeringDict['yloc'],
↪BeringDict['xmax']]])+360
yy = np.array([Y[BeringDict['yloc'],BeringDict['xmin']],Y[BeringDict['yloc'],
↪BeringDict['xmax']]])
#print(xx)
```

```
#print(yy)

map.plot(xx,yy,linewidth=2,color='r')

plt.figure()
plt.plot(x[BeringDict['xmin']:BeringDict['xmax']]+360,transport[BeringDict['yloc'],
↪BeringDict['xmin']:BeringDict['xmax']])
bering_transport = np.sum(transport[BeringDict['yloc'],BeringDict['xmin']:BeringDict[
↪'xmax']])

print(bering_transport)
```

```
<xarray.DataArray 'ty_trans' ()>
array(1.0250125366146676)
Coordinates:
    time      float64 3.591e+04
    yu_ocean  float64 65.75
```

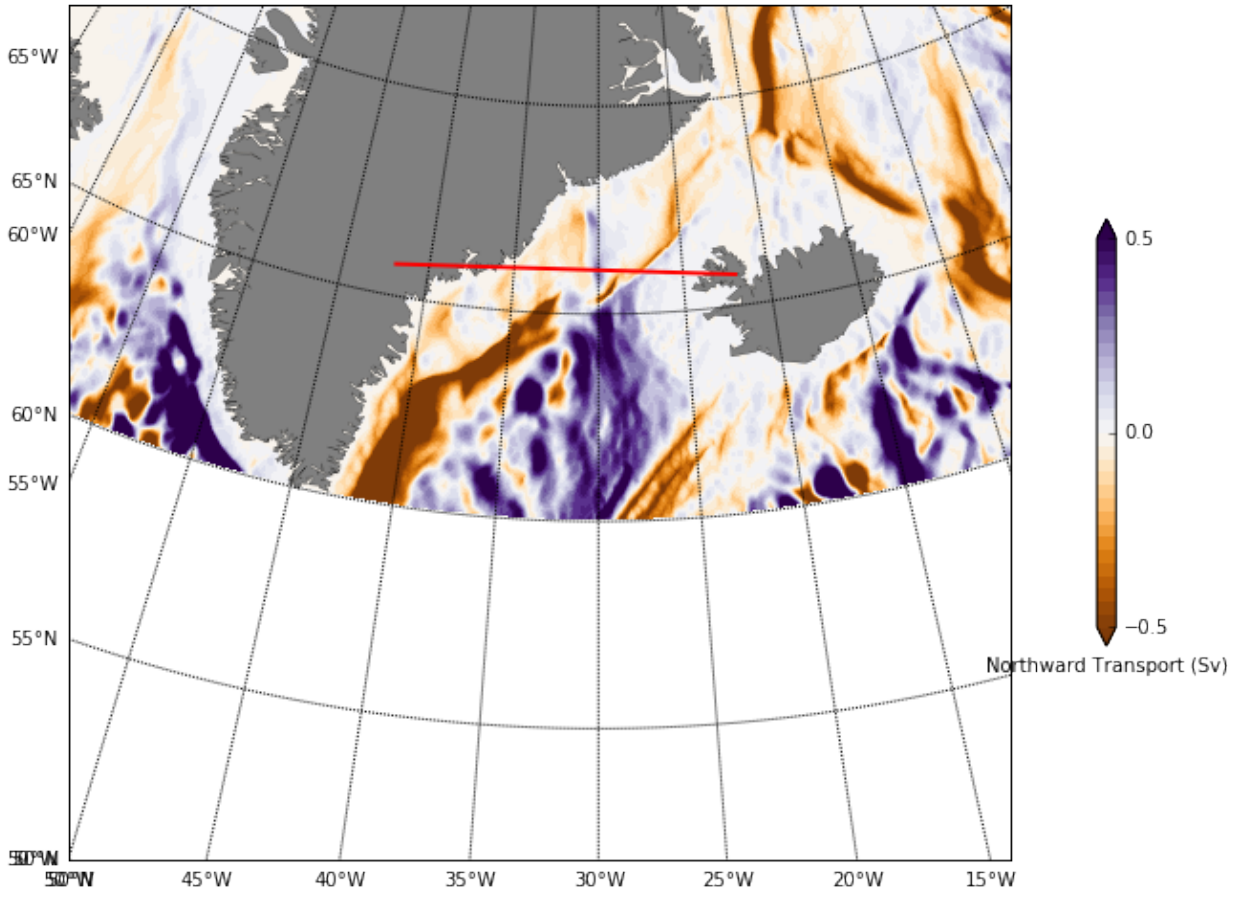


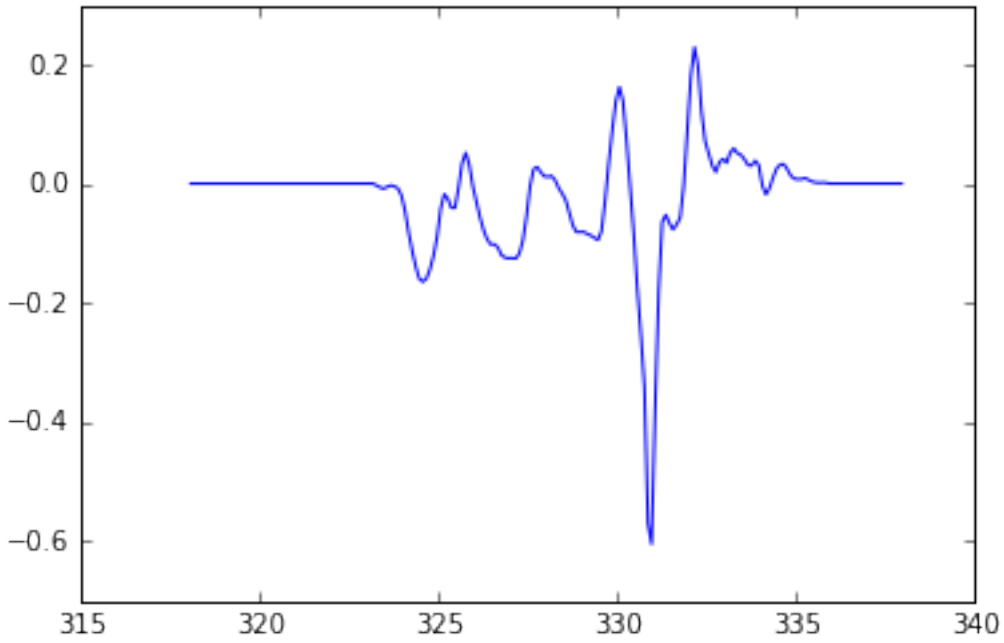



```
# Denmark Strait
# load ty_trans from ocean.nc
OceanFile = os.path.join(DataDir,expts[1],'output383/ocean.nc')
dsx = xr.open_dataset(OceanFile, decode_times=False, engine='netcdf4')
ty_trans = dsx.ty_trans[0,:,2000:,2000:3000]
x = dsx.xt_ocean[2000:3000]
y = dsx.yu_ocean[2000:]

# vertically integrate
transport = ty_trans.sum(axis=0)

# plot local regions around Iceland
plt.figure(figsize=(16,8))
levels = np.linspace(-0.5,0.5,31)
map = Basemap(llcrnrlon=-50,llcrnrlat=50,urcrnrlon=0,urcrnrlat=70,projection='stere',
↪resolution='l',lon_0=-30,lat_0=60)
map.drawcoastlines(linewidth=0.25)
map.fillcontinents(color='gray',lake_color='gray')
# draw parallels and meridians.
map.drawparallels(np.arange(-80.,81.,5.),labels=[True,False,False,True])
map.drawmeridians(np.arange(-180.,181.,5.),labels=[True,False,False,True])
xv, yv = np.meshgrid(x, y)
X, Y = map(xv,yv)
map.contourf(X,Y,transport, cmap=plt.cm.PuOr, levels=levels, extend='both') #CMRmap_r,
↪ gist_stern_r
cb = plt.colorbar(orientation='vertical',shrink = 0.5,ticks=[-1, -0.5, 0, 0.5, 1 ])
cb.ax.set_xlabel('Northward Transport (Sv)')

# pick exact location and sum for each strait
DenmarkDict = {'yloc':2125,'xmin':2380,'xmax':2580}
xx = np.array([X[DenmarkDict['yloc']-2000,DenmarkDict['xmin']-2000], X[DenmarkDict[
↪'yloc']-2000,DenmarkDict['xmax']-2000]])+360
yy = np.array([Y[DenmarkDict['yloc']-2000,DenmarkDict['xmin']-2000],Y[DenmarkDict[
↪'yloc']-2000,DenmarkDict['xmax']-2000]])
```

```
#print(xx)
#print(yy)

map.plot(xx,yy,linewidth=2,color='r')

plt.figure()
plt.plot(x[DenmarkDict['xmin']-2000:DenmarkDict['xmax']-2000]+360,
↪transport[DenmarkDict['yloc']-2000,DenmarkDict['xmin']-2000:DenmarkDict['xmax']-
↪2000])
denmark_transport = np.sum(transport[DenmarkDict['yloc']-2000,DenmarkDict['xmin']-
↪2000:DenmarkDict['xmax']-2000])

print(denmark_transport)
```

```
<xarray.DataArray 'ty_trans' ()>
array(-4.258176312495108)
Coordinates:
    time      float64 3.591e+04
    yu_ocean  float64 65.75
```





Temperature Salinity Plots

```
import logging
logging.captureWarnings(True)
logging.getLogger('py.warnings').setLevel(logging.ERROR)
```

```
import numpy as np
import matplotlib.pyplot as plt
from netCDF4 import Dataset
import pandas as pd
import xarray as xr

import datashader as ds
import datashader.transfer_functions as tf
from datashader.colors import Hot

import dask.array as da
import dask.dataframe as dd
import dask
from dask.diagnostics import ProgressBar
from dask.cache import Cache

from datashader.bokeh_ext import InteractiveImage
import bokeh.plotting as bp
```

```
ProgressBar().register()
Cache(9e9).register()
```

Let's compare different ways of plotting TS diagrams.

```
filename = '/g/data1/v45/mom01_comparison/KDS75/output165/ocean.nc'
```

Loading with netCDF4 and forming a pandas dataframe

```
dataset = Dataset(filename)
```

```
CPU times: user 12 ms, sys: 0 ns, total: 12 ms
Wall time: 36.9 ms
```

```
dataset.variables['age_global']
```

```
<class 'netCDF4._netCDF4.Variable'>
float32 age_global(time, st_ocean, yt_ocean, xt_ocean)
    long_name: Age (global)
    units: yr
    valid_range: [  0.00000000e+00   1.00000002e+20]
    missing_value: -1e+20
    _FillValue: -1e+20
    cell_methods: time: mean
    time_avg_info: average_T1,average_T2,average_DT
    coordinates: geolon_t geolat_t
    standard_name: sea_water_age_since_surface_contact
unlimited dimensions: time
current shape = (1, 75, 2700, 3600)
filling on
```

```
temp = dataset.variables['temp'][:].compressed()
salt = dataset.variables['salt'][:].compressed()
age_global = dataset.variables['age_global'][:].compressed()
```

```
CPU times: user 53.5 s, sys: 6.35 s, total: 59.9 s
Wall time: 59.9 s
```

```
df = pd.DataFrame({'temp': temp, 'salt' : salt, 'age_global' : age_global},␣
↪copy=False)
```

```
CPU times: user 2.24 s, sys: 980 ms, total: 3.22 s
Wall time: 3.21 s
```

temp and salt are numpy MaskedArrays .compressed() returns a 1d array where the masked elements have been removed.

```
# Default plot ranges:
y_range = (-2, 32) # temperature
x_range = (10, 40) # salinity
```

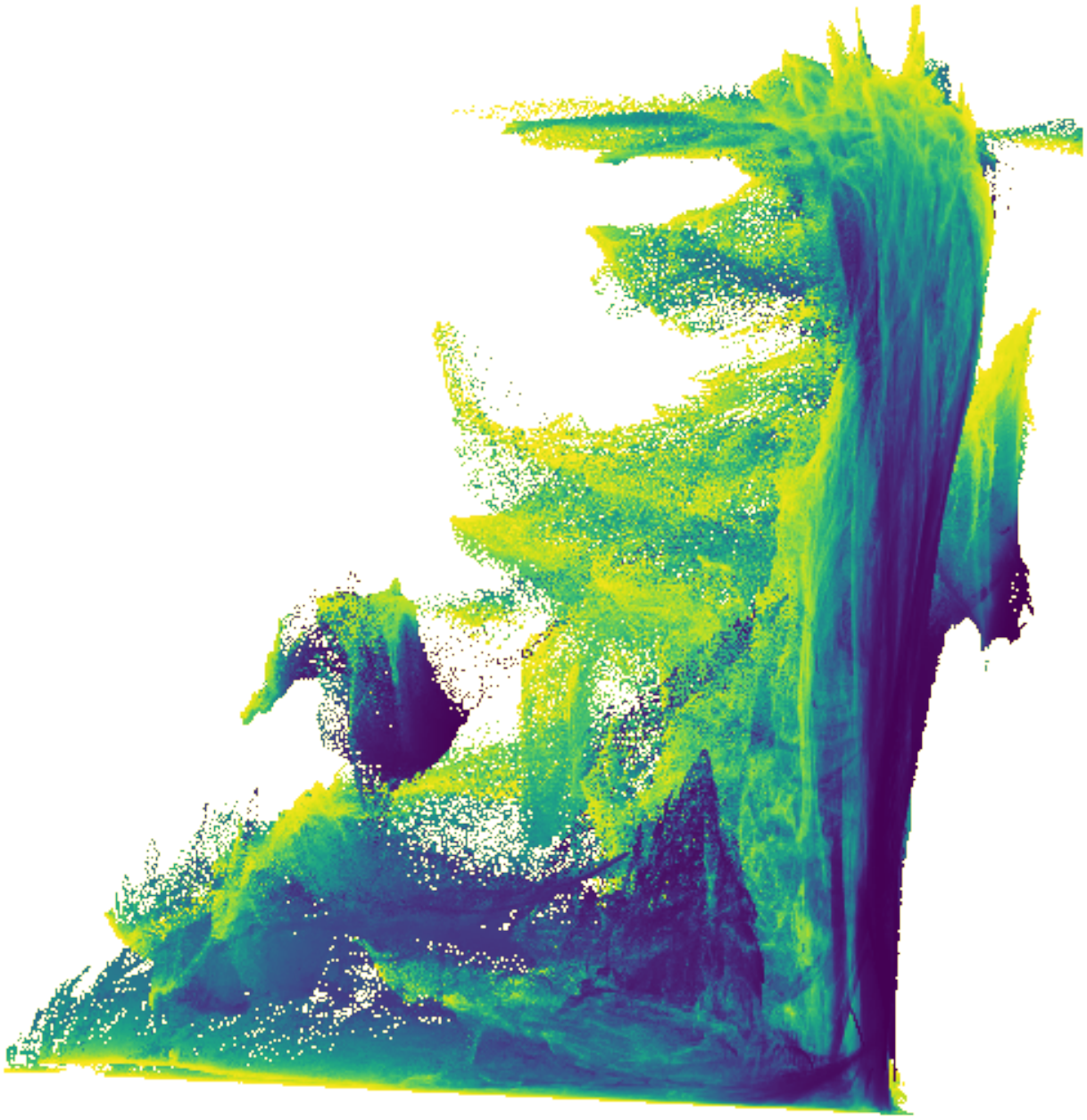
```
def create_image(x_range=x_range, y_range=y_range, w=500, h=500):
    cvs = ds.Canvas(x_range=x_range, y_range=y_range, plot_height=h, plot_width=w)
    agg = cvs.points(df, 'salt', 'temp', ds.mean('age_global'))
    return tf.shade(agg)
```

```
    cvs = ds.Canvas(plot_width=500, plot_height=500,x_range=x_range, y_range=y_range)
agg = cvs.points(df, 'salt', 'temp')
```

```
CPU times: user 10.9 s, sys: 4 ms, total: 11 s
Wall time: 10.9 s
```

```
img = tf.shade(agg, cmap=plt.cm.viridis_r)
img
```

```
/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/matplotlib/colors.
↪py:494: RuntimeWarning: invalid value encountered in less
  cbook._putmask(xa, xa < 0.0, -1)
```



```
bp.output_notebook()

def base_plot(tools='pan,wheel_zoom,box_zoom,reset'):
    p = bp.figure(tools=tools, plot_width=500, plot_height=500,
        x_range=x_range, y_range=y_range, outline_line_color=None,
        min_border=0, min_border_left=0, min_border_right=0,
        min_border_top=0, min_border_bottom=0)
    p.xgrid.grid_line_color = None
    p.ygrid.grid_line_color = None
    p.xaxis.axis_label = 'Salinity'
    p.yaxis.axis_label = 'Potential Temperature'
    return p

p = base_plot()
InteractiveImage(p, create_image)
```

Use dask.dataframe instead of pandas dataframe

```
    dataset = Dataset(filename)

temp = dataset.variables['temp'][:].compressed()
salt = dataset.variables['salt'][:].compressed()
age_global = dataset.variables['age_global'][:].compressed()
```

```
CPU times: user 1min 21s, sys: 10.1 s, total: 1min 31s
Wall time: 1min 31s
```

```
df_dd = dd.from_array(np.array([temp, salt]).T, columns=['temp', 'salt'])
```

```
CPU times: user 9.26 s, sys: 16 ms, total: 9.27 s
Wall time: 40.2 s
```

```
Exception ignored in: <Finalize object, dead>
Traceback (most recent call last):
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/util.py", line␣
↪186, in __call__
    res = self._callback(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪535, in _terminate_pool
    cls._help_stuff_finish(inqueue, task_handler, len(pool))
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪520, in _help_stuff_finish
    inqueue._rlock.acquire()
KeyboardInterrupt
```

```
cvs = ds.Canvas(plot_width=500, plot_height=500,x_range=x_range, y_range=y_range)
agg = cvs.points(df_dd, 'salt', 'temp')
```

```
[########################################] | 100% Completed |  0.1s
[########################################] | 100% Completed | 17.8s
```

Although the dask dataframe is using threads, there is no speed up.

Load data using dask

```
dask.set_options(get=dask.multiprocessing.get, num_workers=4)
```

```
<dask.context.set_options at 0x7f6140123588>
```

```
cvs.points
```

```
<bound method Canvas.points of <datashader.core.Canvas object at 0x7f61e0da8eb8>>
```

```
cvs = ds.Canvas(plot_width=500, plot_height=500,x_range=x_range, y_range=y_range)
agg = cvs.points(df_dd, 'salt', 'temp')
```

```
[                                        ] | 0% Completed | 31.5s
```

```
Process ForkPoolWorker-1:
Process ForkPoolWorker-4:
Process ForkPoolWorker-2:
Process ForkPoolWorker-3:
Process ForkPoolWorker-13:
Process ForkPoolWorker-14:
Process ForkPoolWorker-10:
Process ForkPoolWorker-9:
Process ForkPoolWorker-15:
Process ForkPoolWorker-11:
Process ForkPoolWorker-12:
Traceback (most recent call last):
Traceback (most recent call last):
Traceback (most recent call last):
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
Process ForkPoolWorker-16:
Traceback (most recent call last):
Traceback (most recent call last):
Traceback (most recent call last):
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
Traceback (most recent call last):
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
```

```
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
Traceback (most recent call last):
```

```
[                                        ] | 0% Completed | 31.6s
```

```
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
Traceback (most recent call last):
Traceback (most recent call last):
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
```

```
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
Traceback (most recent call last):
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
```

```
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
Traceback (most recent call last):
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 343, in get
    res = self._reader.recv_bytes()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/connection.py",
↪ line 216, in recv_bytes
    buf = self._recv_bytes(maxlength)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
```

```
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/connection.py",
↪ line 407, in _recv_bytes
    buf = self._recv(4)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/connection.py",
↪ line 379, in _recv
    chunk = read(handle, remaining)
```

```
---------------------------------------------------------------------------

KeyboardInterrupt                         Traceback (most recent call last)

<ipython-input-91-58f21cd65646> in <module>()
      1 cvs = ds.Canvas(plot_width=500, plot_height=500,x_range=x_range, y_range=y_
↪range)
----> 2 agg = cvs.points(df_dd, 'salt', 'temp')


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/datashader/core.py in␣
↪points(self, source, x, y, agg)
    140         if agg is None:
    141             agg = count()
--> 142         return bypixel(source, self, Point(x, y), agg)
    143
    144     def line(self, source, x, y, agg=None):


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/datashader/core.py in␣
↪bypixel(source, canvas, glyph, agg)
    310     agg : Reduction
    311     """
--> 312     dshape = discover(source)
    313     if not istabular(dshape):
    314         raise ValueError("source must be tabular")


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/multipledispatch/
↪dispatcher.py in __call__(self, *args, **kwargs)
    162             self._cache[types] = func
    163         try:
--> 164             return func(*args, **kwargs)
    165
```

```
    166         except MDNotImplementedError:


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/odo/backends/dask.py in␣
↪discover_dask_dataframe(df)
     25 @discover.register(dd.DataFrame)
     26 def discover_dask_dataframe(df):
---> 27     return var * discover(df.head()).measure
     28
     29


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/dask/dataframe/core.py␣
↪in head(self, n, npartitions, compute)
    793
    794         if compute:
--> 795             result = result.compute()
    796         return result
    797


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/dask/base.py in␣
↪compute(self, **kwargs)
     92             Extra keywords to forward to the scheduler ``get`` function.
     93         """
---> 94         (result,) = compute(self, traverse=False, **kwargs)
     95         return result
     96


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/dask/base.py in␣
↪compute(*args, **kwargs)
    199     dsk = collections_to_dsk(variables, optimize_graph, **kwargs)
    200     keys = [var._keys() for var in variables]
--> 201     results = get(dsk, keys, **kwargs)
    202
    203     results_iter = iter(results)


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/dask/multiprocessing.py␣
↪in get(dsk, keys, num_workers, func_loads, func_dumps, optimize_graph, **kwargs)
     84         result = get_async(pool.apply_async, len(pool._pool), dsk3, keys,
     85                            get_id=_process_get_id,
---> 86                            dumps=dumps, loads=loads, **kwargs)
     87     finally:
     88         if cleanup:


/home/v45/miniconda3/envs/cosima/lib/python3.6/site-packages/dask/async.py in get_
↪async(apply_async, num_workers, dsk, result, cache, get_id, raise_on_exception,␣
↪rerun_exceptions_locally, callbacks, dumps, loads, **kwargs)
    489         # Main loop, wait on tasks to finish, insert new ones
    490         while state['waiting'] or state['ready'] or state['running']:
--> 491             key, res_info = queue.get()
    492             res, tb, worker_id = loads(res_info)
    493             if isinstance(res, BaseException):
```

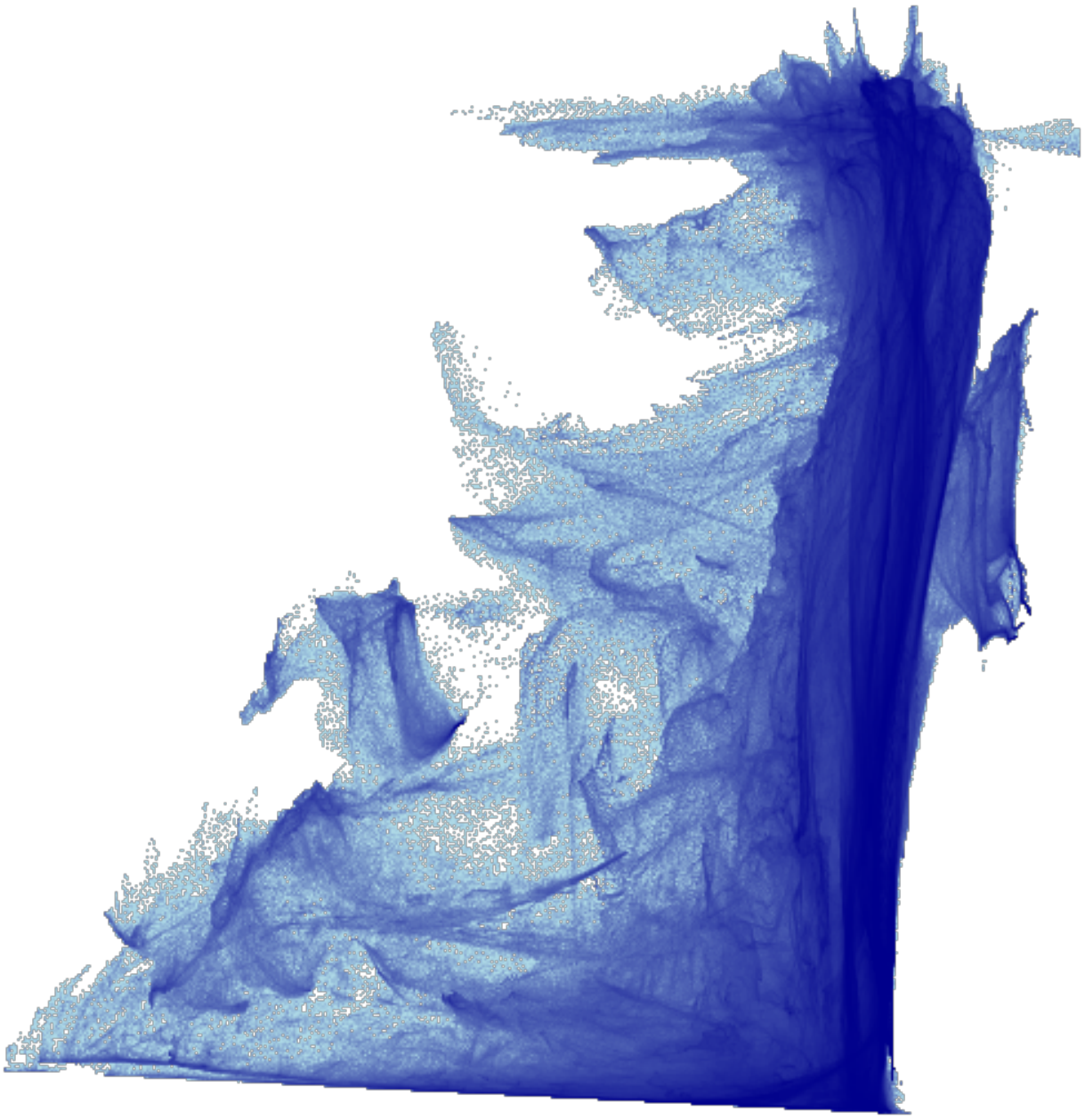
```
/home/v45/miniconda3/envs/cosima/lib/python3.6/queue.py in get(self, block, timeout)
    162             elif timeout is None:
    163                 while not self._qsize():
--> 164                     self.not_empty.wait()
    165             elif timeout < 0:
    166                 raise ValueError("'timeout' must be a non-negative number")


/home/v45/miniconda3/envs/cosima/lib/python3.6/threading.py in wait(self, timeout)
    293         try:    # restore state no matter what (e.g., KeyboardInterrupt)
    294             if timeout is None:
--> 295                 waiter.acquire()
    296                 gotit = True
    297             else:


KeyboardInterrupt:
```

```
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
Process ForkPoolWorker-5:
Process ForkPoolWorker-6:
Process ForkPoolWorker-7:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
KeyboardInterrupt
Traceback (most recent call last):
Traceback (most recent call last):
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
```

```
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
KeyboardInterrupt
Traceback (most recent call last):
KeyboardInterrupt
KeyboardInterrupt
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
Process ForkPoolWorker-8:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
Traceback (most recent call last):
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
```

```
    return self._semlock.__enter__()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 249, in _bootstrap
    self.run()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/process.py",␣
↪line 93, in run
    self._target(*self._args, **self._kwargs)
KeyboardInterrupt
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/pool.py", line␣
↪108, in worker
    task = get()
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/queues.py",␣
↪line 342, in get
    with self._rlock:
  File "/home/v45/miniconda3/envs/cosima/lib/python3.6/multiprocessing/synchronize.py
↪", line 96, in __enter__
    return self._semlock.__enter__()
KeyboardInterrupt
```

```
tf.shade(agg)
```



```
from dask.distributed import Client, LocalCluster
```

```
cluster.status
```

```
'closed'
```

```
cluster = LocalCluster(n_workers=4, threads_per_worker=1)
```

```
client = Client(cluster)
```

```
client.restart()
```

```
<Client: scheduler='tcp://127.0.0.1:8786' processes=0 cores=0>
```

```
filename = '/g/data1/v45/mom01_comparison/KDS75/output165/ocean.nc'
```

```
dsx = xr.open_dataset(filename, decode_times=False)
```

```
# convert to pandas dataframe
```

```
# convert to pandas dataframe
```

```
df3 = df.merge(df2)
```

```
stacked = dsx.temp.stack(z = ('st_ocean', 'xt_ocean', 'yt_ocean'))
```

```
stacked.z
```

```
    agg = cvs.points(df, 'geolon_t', 'geolat_t', ds.count('temp'))

img = tf.shade(agg, how='log')
img
```

```
img = tf.shade(agg, how='log')
```

```
from dask import delayed

@delayed
def get_data(key):
    ncfile = '/g/data1/v45/mom01_comparison/KDS75_wind/output165/ocean.nc'
    h5 = h5py.File(ncfile, mode='r')
    arr = dd.from_array(h5['/'+key][:].ravel(), columns = [key])
    h5.close()
    return arr

temp = get_data('temp')
print('temp loaded')
salt = get_data('salt')
print('salt loaded')

#arr = delayed(np.hstack)((temp, salt))
print(arr.shape)
#df = delayed(dd.from_array)( arr, columns = ['temp', 'salt'])
print(df)
```

```
ncfile = '/g/data1/v45/mom01_comparison/KDS75_wind/output165/ocean.nc'
dsx = xr.open_dataset(ncfile, decode_times=False, engine='h5netcdf')
```

```
from dask import delayed

@delayed
def get_data(key):
    ncfile = '/g/data1/v45/mom01_comparison/KDS75_wind/output165/ocean.nc'
    dsx = xr.open_dataset(ncfile, decode_times=False, engine='h5netcdf')
    arr = dsx.variables(key).values.ravel()
```

```
    dsx.close()
    return arr

df = dd.from_array( np.array( [ dsx.temp.values.ravel(), dsx.salt.values.ravel()]).T,
              columns=['temp', 'salt'])
```

```
dsx = xr.open_dataset(ncfile, decode_times=False, engine='h5netcdf', chunks={'st_ocean
↪':1, 'sw_ocean':1})
```

```
df = dd.from_array( np.array( [ dsx.temp.values.ravel(), dsx.salt.values.ravel()]).T,
              columns = ('temp', 'salt'))
```

```
df = delayed(dd.merge)(temp, salt)
```

```
df.visualize()
```

```
df2 = df.compute()
```

```
ncfile = '/g/data1/v45/mom01_comparison/KDS75_wind/output165/ocean.nc'
dd.read_hdf(ncfile, 'temp')
```

CHAPTER 5

Indices and tables

- genindex
- search