May 19, 2015

PyAverager

A package used for computing averages from climate model output.

Authors: Sheri Mickelson, Kevin Paul, and John Dennis
Version: 0.9.1

Comments and feedback: mickelso@ucar.edu

What the PyAverager Can Do

The PyAverager can create the climatology files needed by the AMWG, OMWG, Land, and Ice CESM diagnostic packages (the full list of averages is defined within the 'Specification' section). It is able to compute averages from previously generated averages (such as monthly averages for season averages) or from scratch. The PyAverager can operate on monthly time-slice and time-series files that exist in the same directory.

Dependencies

The PyAverager depends on the PyNIO, mpi4py, and PyTools packages to be installed on your system. PyNIO is needed for NetCDF file I/O. mpi4py and PyTools are needed for the parallel communication, though it is possible to run the PyAverager in serial mode without mpi4py.

If you are not running on a CESM supported machine, installation information can be found at:

- PyNIO: https://www.pyngl.ucar.edu/Download/
- Mpi4py: http://mpi4py.scipy.org/
  (git clone https://bitbucket.org/mpi4py/mpi4py.git)
- PyTools: https://www2.cisl.ucar.edu/tdd/asap/parallel-python-tools-post-processing-climate-data
  (svn co https://proxy.subversion.ucar.edu/pubasap/pyTools/tags/v0.3/ or git clone https://github.com/NCAR-CISL-ASAP/ASAPPyTools)

CESM Supported Machine Information (steps to take to get these packages into your path):

Yellowstone:
Add the following to the top of your bsub script or execute them on the command line and then type 'module save' to always keep them in your environment:

- module load python
- module load all-python-libs
- module load asaptools
  (If asaptools fails to load, it has since been added to 'all-python-libs' module)

tukey.alcf.anl.gov:

- Edit your $HOME/.soft.tukey file
  - Add (before @default):
    - @PyNIO-1.4.1-anaconda
    - @PyNGL-1.4.0-anaconda
    - +mpich2-1.4.1p1
    - +anaconda
- Execute 'resoft'
- See below to install ASAP PyTools

edison.nersc.gov:
Follow the Yellowstone instructions, but load:

- module load python_base/2.7.3
- module load mpi4py/1.3.1
- module load numpy
- Add /global/u1/m/mickelso/python_libs/lib/python2.7/site-packages/PyNIO/ to your PYTHONPATH env variable
- See below to install ASAP PyTools

How To Install ASAP PyTools

- git clone https://github.com/NCAR-CISL-ASAP/ASAPPyTools pytools
- cd into pytools
- python setup.py install --user

Building and Installing the PyAverager

Check out the source code `(3 options):`

- svn co https://proxy.subversion.ucar.edu/pubasap/pyAverager/tags/v0.9.1/
- git clone https://github.com/sherimickelson/pyAverager
- pip install pyAverager --user

If checking out the code via svn or git, you must install the pyAverager (the pip method will install the package for you).

- $ cd PyAverager

- $ python setup.py install –user

Make sure the install location is added to your $PYTHONPATH

- Type 'python' in a terminal to get the interactive terminal.
- Type '`from pyaverager import PyAverager, specification`'
- You will get an error if it is not in your path

To install documentation, run:

- $ doxygen Doxyfile
  The documentation will be created in the apidocs directory.

Running the Examples

Running the examples on Yellowstone:
(For other machines, you will need to create a queue submission script similar to examples/ runAvg_mpi.csh)

- $ cd examples
- Open runAvg_mpi.csh for editing
- Set the correct project number to run under
- Select an example to run (control_*.py)
- Edit the control_*.py script you would like to run
  (See the 'Specifier' section below for more details on editing the control script)
- Run
  - $ bsub < runAvg_mpi.csh

Specification

The PyAverager is a python library that is referenced from another python script. In order to run the PyAverager, you need to specify parameters so the average knows what types of averages to compute, input/output locations, and any averaging options you would like to add. The 'example' directory contains several 'control_.py' files that you can use as templates. You can copy one of these scripts and modify the top section to fit your data.

CESM naming conventions that the PyAverager follows by default:

Slice: $CASE.$comp.$stream.$year-$month.nc

Series: $CASE.$comp.$stream.$var.$year1$month1-$year2$month2.nc
(If your file names do not match this pattern, you will need to pass the file_pattern variable to the specifier)

The table below lists the types of averages the PyAverager can compute.

Average Option	Description	Output Name	Can be Weighted?	Can Be Created As a Dependency?
ya	Yearly Average	$CASE.$YEAR.nc	Yes	No
tavg	Ocn average across years	tavg.$Year1-$Year2.nc	Yes	Yes
annall	Land model, annual averages concat together	$CASE_ANN_ALL.nc	Yes	Yes
moc	Ocn MOC file	$CASE_moc.nc	Yes	Yes
hor.meanyr	Ocn hor.mean year file	$REG_hor.meanyr.$YEAR.nc	Yes	Yes
hor.meanConcat	Ocn, concat of hor.meayr	$REG_hor_mean_hor.meanConcat. $CASE.$Year1-$Year2.nc	Yes	Yes
mocm	Ocn MOCM	$CASE_mocm.nc	No	No
ann	Annual Average	$CASE_ANN_climo.nc	Yes	Yes
djf	Winter Average	$CASE_DJF_climo.nc	Yes	Yes
mam	Spring Average	$CASE_MAM_climo.nc	Yes	Yes
jja	Summer Average	$CASE_JJA_climo.nc	Yes	Yes
son	Fall Average	$CASE_SON_climo.nc	Yes	Yes
jan	January Average	$CASE_01_climo.nc	Yes	Yes
feb	February Avg	$CASE_02_climo.nc	Yes	Yes
mar	March Average	$CASE_03_climo.nc	Yes	Yes
apr	April Average	$CASE_04_climo.nc	Yes	Yes
may	May Average	$CASE_05_climo.nc	Yes	Yes
jun	June Average	$CASE_06_climo.nc	Yes	Yes
jul	July Average	$CASE_07_climo.nc	Yes	Yes
aug	August Average	$CASE_08_climo.nc	Yes	Yes
sep	Sept Average	$CASE_09_climo.nc	Yes	Yes
oct	Oct Average	$CASE_10_climo.nc	Yes	Yes
nov	Nov Average	$CASE_11_climo.nc	Yes	Yes
dec	Dec Average	$CASE_12_climo.nc	Yes	Yes
mavg	Concat of all monthly averages	mavg.$Year1-$Year2.nc	No	Yes
mons	Lnd, concat of monthly averages	$CASE_MONS_climo.nc	No	Yes
jfm	Ice Winter Avg	$CASE_jfm_climo.nc	Not Now	Yes
fm	Ice Feb & Mar Avg	$CASE_fm_climo.nc	Not Now	Yes
amj	Ice Spring Avg	$CASE_amj_climo.nc	Not Now	Yes
jas	Ice Summer Avg	$CASE_jas_climo.nc	Not Now	Yes
ond	Ice Fall Avg	$CASE_ond_climo.nc	Not Now	Yes
on	Ice Oct & Nov Avg	$CASE_on_climo.nc	Not Now	Yes
preproc	Ice pre proc file	ice_vol$CASE_$Year1-$Year2.nc	Not Now	Yes

Can be created as a dependency? option:
The averages that are listed in the above table as being able to create averages as dependencies have the ability to use previously calculated averages to calculate a new average. To use this option, append 'dep_' in front of the average name (ie, 'dep_jja'). Without 'dep_', a jja average would loop over all June, July, and August values within the year ranges and create an average. With 'dep_', the PyAverager will create and output a June average, July average, and August average. Then it will open these average files and average these values to create the 'jja' average file. In most cases, it is faster to run with 'dep_', but it should be pointed out that the answers between using and not using the 'dep_' option will differ due to order of operation.

Specifier Arguements
See examples/control.py for how to set all available options to send to the create_specifier function.

Variables that must be passed to the specification.create_specifier class:

- in_directory: directory where the input data is located
- out_directory: directory where the output will be produced
- prefix: the case name, plus component name (ie. b40.20th.track1.1deg.006.cam2.h0)
- suffix: the end of the input file names (usually nc)
- date_pattern: `'yyyymm-yyyymm'`
- avg_list: a list of averages to compute DEFAULT = Empty List
  Format: ['ya:1850','mavg:1850:1890'] ya is the only average to take one year. All other averages expect a start year and end year separated by a colon. The available average choices are listed in the above table.

Variables that are mandatory for the Ice and Ocean Diags:

- The following variables are used by the Ocean Model Diags for the hor.meanConcat file creatation:
  - mean_diff_rms_obs_dir: directory that contains the obervartion files needed to calculate the hor.mean.Concat file (Ocean Model).
  - region_obs_file_suffix: the suffix of region obs files found in the mean_diff_rms_obs_dir directory
  - region_nc_var: variable name the contains the region mask information (Ocean Model)
  - regions: regions to create files for (ie`[1:'Sou',2:'Pac']`) region int that corresponds to the region_mask, region name.
  - region_wgt_var: variable name that contains the region weight info
  - obs_file: observation file (contains the region_nc_var and region_wgt_var)
  - obs_dir: directory where the obs_file is located in
- The following variables are used by the Ice Model Diags for the Pre_Proc file:
  - ice_obs_file: a netCDF file that contains area/weight information
  - ncl_location: the location of the ncl script used to create the reg_file (usually provided with this source code in pyaverager/ directory

- reg_file: the name of the netcdf file that contains the region mask information. If it does not exist, it will be created for you.

Optional variables that can be passed to the specification.create_specifier class:

- ncformat: either 'netcdf4c' (netcdf4 compressed (lev=1)), 'netcdf4' (netcdf classic), and 'netcdf' (netcdf3 classic) DEFAULT = 'netcdf4c'
- file_pattern: needed for non-cesm data
  For file name: `tasmax_Amon_GFDL-FLORB01_FLORB01-P1-ECDA-v3.1-011980_r10i1p1_198001-198012.nc`
  Use: `['$var','_','$prefix','_','$m_id','_','$date_pattern','.','$suffix']`
- hist_type: either 'slice' or 'series' DEFAULT = 'slice'
- m_id: experiment/or other unique id (can be used to id ensemble members)
- weighted: Boolean to weight averages (when available, see about table) DEFAULT = False
- split: Are the files split between lat coordinates (used in cice series files) DEFAULT = False
- split_files: strings differentiating the different pieces DEFAULT = 'null'
- split_orig_size: list of lat/lon names and their original full size DEFAULT = 'null'
- varlist: ['a','list','of','vars','to','avg'] DEFAULT = Full list
- clobber: If a user specified average exists on disk, delete if set to true. DEFAULT=False
- serial: run in serial or parallel mode DEFAULT=FALSE (parallel mode)
- main_comm: the simple_comm object. If one isn't passed in, one will be initialized for you.

To generate the obs file needed for the Ocean hor.meandiff calculation:
(or yellowstone users can copy/use files in /glade/p/work/mickelso/PyAvg-OMWG-obs/obs)

Required:

- From omwg obs_data:
  PHC2_TEMP_gx1v6_ann_avg.nc and SALT PHC2_SALT_gx1v6_ann_avg.nc
- POP history file

Directions:

- Copy PHC2_TEMP_gx1v6_ann_avg.nc to obs.nc
- ncks -A -v SALT PHC2_SALT_gx1v6_ann_avg.nc obs.nc
- ncks -A -v TAREA,REGION_MASK a_pop_history_file.nc obs.nc
- ncrename -O -d X,nlon -d Y,nlat -d depth,z_t obs.nc
- ncatted -a _FillValue,TEMP,c,f,-99. obs.nc
- ncatted -a _FillValue,SALT,c,f,-99. obs.nc
- ncatted -a missing_value,TLAT,d,, obs.nc
- ncatted -a missing_value,TLONG,d,, obs.nc
- ncatted -a _FillValue,TLAT,d,, obs.nc

- ncatted -a _FillValue,TLONG,d,, obs.nc
- ncatted -a _FillValue,TAREA,c,f,-99. obs.nc
- ncatted -a _FillValue,TAREA,m,f,1.0e36 obs.nc
- ncatted -a _FillValue,TAREA,m,f,-99 obs.nc
- ncatted -a _FillValue,,m,f,-99 obs.nc
- ncatted -a _FillValue,TAREA,m,f,-99 obs.nc
- ncatted -a _FillValue,TAREA,o,f,-99 obs.nc
- ncatted -a _FillValue,REGION_MASK,o,i,99 obs.nc

To generate the regional obs files needed for the Ocean hor.meandiff calculation:
(or yellowstone users can copy/use files in /glade/p/work/mickelso/PyAvg-OMWG-obs/obs)
Required:

- From omwg obs_data: PHC2_TEMP_gx1v6_ann_avg.nc and SALT PHC2_SALT_gx1v6_ann_avg.nc
- POP history file

Directions:

- Copy PHC2_TEMP_gx1v6_ann_avg.nc to obs.nc
- ncks -A -v SALT PHC2_SALT_gx1v6_ann_avg.nc obs.nc
- ncks -A -v TAREA,REGION_MASK a_pop_history_file.nc obs.nc
- ncrename -O -d X,nlon -d Y,nlat -d depth,z_t obs.nc
- ncatted -a _FillValue,TEMP,c,f,-99. obs.nc
- ncatted -a _FillValue,SALT,c,f,-99. obs.nc
- ncatted -a missing_value,TLAT,d,, obs.nc
- ncatted -a missing_value,TLONG,d,, obs.nc
- ncatted -a _FillValue,TLAT,d,, obs.nc
- ncatted -a _FillValue,TLONG,d,, obs.nc
- For regions:
    - ncwa -m REGION_MASK -T eq -M <reg_number> -w TAREA -a nlon,nlat -v TEMP,SALT obs.nc Arc_hor_mean_obs.nc
    - <reg_number> Table

Sou	Pac	Ind	Atl	Lab	Gin	Arc	Hud
1	2	3	6	8	9	10	11

- For Glo:
    - ncwa -m REGION_MASK -T gt -M 0 -w TAREA -a nlon,nlat -v TEMP,SALT obs.nc Glo_hor_mean_obs.nc

PyAverager Error Codes

errors 1-19: average list errors
1: Listed average is not in the know average list

2: Average cannot be created with dependencies
3: Average must list only one year
4: Average must have a start year and an end year
5: Date ranges are inconsistent and cannot run this average with dependencies

errors 20-39: input file problems
20: Cannot find the file (triggered in three different checks points)
21: Missing files to calculate DJF. You need either the previous December or the January and February from last year+1
22: Time series files are split, but the dates between them are not contiguous (triggered in two different checks points)
23: A date was found within two different time series files. Not sure which to use.