Memory Dumping on Cheyenne with a Simple Script

December 6th 2017

Problem Description: I am running a simple script that correlates an ensemble of 34 1-D time series with another set of 34 1-D time series (low memory). When running on a node on Cheyenne, the script fails due to consuming too much memory and core dumps.

*#!/bin/bash*

*#PBS -A P93300670*

*#PBS -N global\_variability*

*#PBS -l walltime=01:00:00*

*#PBS -M riley.brady@colorado.edu*

*#PBS -q regular*

*#PBS -l select=1:ncpus=18*

*#PBS -m abe*

**Test 1:** Run in interactive mode (1 node, 6 CPUs). See where the code fails.

* Climate index data is ~3.4MB
* Core dumping is occurring when trying to open the Canary Current time series. **This file is only 322K**.
* Seems to be an issue with opening certain filetypes… a Python-netCDF issue. A specific help thread suggests installing h5netcdf. THIS SOLVES!

Top of Form

Bottom of Form

152 lines (143 sloc)  5.89 KB

|  |  |
| --- | --- |
|  | """ |
|  | Area Weighted EBUS Correlation |
|  | ---------------------------- |
|  | Author: Riley X. Brady |
|  | Date: Oct 11, 2017 |
|  |  |
|  | This script will correlate the area-weighted residuals time series for a given |
|  | EBUS with a climate index time series (from its corresponding ensemble member). |
|  | This is an updated script from the old version that used to use numpy arrays. |
|  | This will use xarray to its fully capacity and output an entire ensemble of |
|  | correlations as a single netCDF file. |
|  |  |
|  | E.g. you can correlate the entire California Current FG\_ALT\_CO2 residuals with |
|  | the NPGO index. You can also specify whatever lag and smoothing necessary. |
|  |  |
|  | NOTE: To compare with an EOF of CO2 flux, just use EOF1, EOF2, or EOF3 for the |
|  | VARY argument. However, beware that this is only operational for FG\_ALT\_CO2 |
|  | EOFs currently. |
|  |  |
|  | NOTE: There is probably a way to do this with "apply" but I can't figure out |
|  | how. It isn't as simple as doing .groupby('ensemble') and then calling |
|  | ds.x and ds.y in the apply function. |
|  |  |
|  | NOTE: This script is also written to handle PDO, ENSO, AMO, etc. from the |
|  | climate diagnostics package as well as NPGO. You will have to add functionality |
|  | for other single time series in the future. |
|  |  |
|  | INPUT 1: Str for EBUS ('CalCS', 'CanCS', 'HumCS', 'BenCS') |
|  | INPUT 2: Predictor climate variable for the residuals |
|  | ('NPGO', 'PDO', 'ENSO', 'AMO', etc.) |
|  | INPUT 3: Dependent variable in the EBU |
|  | (FG\_ALT\_CO2, FG\_CO2, EOF1, EOF2, EOF3) |
|  | INPUT 4: Int for number of months to lag (0 is no lag). |
|  | INPUT 5: Int for how many months to smooth by (0 is no smoothing). |
|  | """ |
|  | import glob |
|  | import sys |
|  | import os |
|  | import numpy as np |
|  | import xarray as xr |
|  | import esmtools as et |
|  |  |
|  | def main(): |
|  | EBU = sys.argv[1] |
|  | VARX = sys.argv[2] |
|  | VARY = sys.argv[3] |
|  | LAG = int(sys.argv[4]) |
|  | SMOOTH = int(sys.argv[5]) |
|  | print("Working on " + VARX + " regressions over the " + EBU + |
|  | " with " + str(LAG) + "mo. lag and " + str(SMOOTH) + |
|  | " mo. smoothing...") |
|  | if VARX == 'NPGO': |
|  | """ |
|  | This was a custom EOF procedure, so the NC files are very different |
|  | from the way Adam Phillips set his up. |
|  | """ |
|  | filepath = '/glade/p/work/rbrady/NPGO/' |
|  | ds\_x = xr.open\_mfdataset(filepath + '\*.nc', concat\_dim='ensemble') |
|  | ds\_x = ds\_x['pc'] |
|  | elif VARX == 'NPH': |
|  | """ |
|  | This is another custom procedure. A simple box over the Northeast |
|  | Pacific that represents anomalies in the standard position of the |
|  | NPH. |
|  | """ |
|  | filepath = '/glade/p/work/rbrady/EBUS\_BGC\_Variability/indices/NPH/' |
|  | filename = 'NPH.full\_ensemble.192001-201512.nc' |
|  | ds\_x = xr.open\_dataset(filepath + filename) |
|  | ds\_x = ds\_x['NPH'] |
|  | else: |
|  | """ |
|  | This assumes the variable can be found in Adam Phillip's climate |
|  | diagnostics output. Need to edit this loading if that's not the |
|  | case. |
|  | """ |
|  | filepath = '/glade/p/work/rbrady/cesmLE\_CVDP/processed/' |
|  | filename = 'cvdp\_detrended\_BGC.nc' |
|  | ds\_x = xr.open\_dataset(filepath + filename) |
|  | ds\_x = ds\_x[VARX.lower()] |
|  | # Load in the Y variable. |
|  | if VARY in ['EOF1', 'EOF2', 'EOF3']: |
|  | filepath = ('/glade/p/work/rbrady/EBUS\_BGC\_Variability/' + |
|  | 'regional\_EOFs/' + EBU + '/FG\_ALT\_CO2/') |
|  | filename = 'FG\_ALT\_CO2.' + EBU + '.EOF.192001-201512.nc' |
|  | ds\_y = xr.open\_dataset(filepath + filename) |
|  | if VARY == 'EOF1': |
|  | ds\_y = ds\_y['pc'].sel(mode=0) |
|  | elif VARY == 'EOF2': |
|  | ds\_y = ds\_y['pc'].sel(mode=1) |
|  | else: |
|  | ds\_y = ds\_y['pc'].sel(mode=2) |
|  | # Remove any trend (should be very slight). |
|  | ds\_y = ds\_y.groupby('ensemble', squeeze=True) \ |
|  | .apply(et.ufunc.remove\_polynomial\_fit) |
|  | else: |
|  | # Load in the co2 flux anomalies. |
|  | filepath = ('/glade/p/work/rbrady/EBUS\_BGC\_Variability/' + VARY + '/' + |
|  | EBU + '/filtered\_output/') |
|  | filename = EBU.lower() + '-' + VARY + '-residuals-AW-chavez-800km.nc' |
|  | ds\_y = xr.open\_dataset(filepath + filename) |
|  | ds\_y = ds\_y[VARY + '\_AW'] |
|  | # Smooth if necessary. |
|  | if SMOOTH != 0: |
|  | ds\_x = ds\_x.rolling(time=SMOOTH).mean().dropna('time') |
|  | ds\_y = ds\_y.rolling(time=SMOOTH).mean().dropna('time') |
|  | # Combine into one dataset. |
|  | ds\_x.name = 'x' |
|  | ds\_y.name = 'y' |
|  | ds = ds\_x.to\_dataset() |
|  | ds['y'] = ds\_y |
|  | # Run the correlation (Here can definitely be improved..) |
|  | m, r, p = ([] for i in range(3)) |
|  | for label, group in ds.groupby('ensemble'): |
|  | """ |
|  | Run a simple correlation/regression, but need to check for all of the |
|  | optional lag and smoothing settings. |
|  | """ |
|  | if LAG == 0: |
|  | M, \_, R, P, \_ = et.stats.linear\_regression(group.x, group.y) |
|  | m.append(M) |
|  | r.append(R) |
|  | p.append(P) |
|  | else: |
|  | M, \_, R, P, \_ = et.stats.linear\_regression(group.x[:-LAG], |
|  | group.y[LAG:]) |
|  | m.append(M) |
|  | r.append(R) |
|  | p.append(P) |
|  | # Set up in dataset. |
|  | ds = xr.Dataset({'m': ('ensemble', m), |
|  | 'r': ('ensemble', r), |
|  | 'p': ('ensemble', p)}) |
|  | print("Finished regional correlations.") |
|  | OUT\_DIR = ('/glade/p/work/rbrady/EBUS\_BGC\_Variability/' + |
|  | 'area\_weighted\_regional\_regressions/' + EBU + '/' + VARY + '/' + |
|  | VARX + '/') |
|  | if not os.path.exists(OUT\_DIR): |
|  | os.makedirs(OUT\_DIR) |
|  | if SMOOTH != 0: # Save with smoothing in file name. |
|  | out\_file = (OUT\_DIR + VARX + '.' + VARY + '.' + EBU + '.smoothed' + |
|  | str(SMOOTH) + '.area\_weighted\_regional\_regression.lag' + |
|  | str(LAG) + '.nc') |
|  | else: |
|  | out\_file = (OUT\_DIR + VARX + '.' + VARY + '.' + EBU + |
|  | '.unsmoothed.area\_weighted\_regional\_regression.lag' + |
|  | str(LAG) + '.nc') |
|  | print("Saving to netCDF...") |
|  | ds.to\_netcdf(out\_file) |
|  |  |
|  | if \_\_name\_\_ == '\_\_main\_\_': |
|  | main() |