Parallel Python Tools Home

Parallel Python Tools for Post-Processing Climate Data:

For use in the the new end-to-end CESM Workflow

PyReshaper

Time-Slice to Time-Series Conversion Tool:

Modern climate model simulations can produce Petabytes of data. Efficient use of this data in post-processing requires converting it from time-slice into time-series files. The PyReshaper is a parallel Python package for performing this conversion quickly and efficiently.

History Time-Slice to Time-Series Converter – Serial NCO

There is a tremendous amount of parallelism to exploit in the conversion process. In our task-parallel approach, each compute rank is responsible for writing one (or more) time-series files. This approach scales up to the number of variables in the dataset.

PyAverager

Computing Climatological Average Files in Parallel:
Computing climatological average files is one metric scientists use to evaluate climate model output. The current process involves serial calls to the NetCDF Operator Tools (NCO) to create average files. An increase in spatial and temporal resolutions has made this process extremely time consuming. In some cases it's not possible to create these files due to memory constraints.

There is a very high amount of parallelism that can be exploited within this computation. Our approach task parallelizes over averages and variables. It also allows for a much smaller memory footprint as compared to the NCO tools.

### PyTools

#### MPI and Timing Wrappers:

The PyTools package contains wrappers to help with two commonly used utilities. The first library contains wrappers around commonly used MPI routines. This package makes it very easy to distribute tasks between MPI ranks and eases the use of several communication strategies. The second library contains wrappers around timer routines and make it easier to keep track of timing within your own code. Both the PyAverager and the PyReshaper use versions of the libraries for timings and MPI communication and are provided within each package. PyTools can also be used separately within your own code to add parallelization and timing capabilities.

### Performance Results from the PyReshaper and the PyAverager
PyReshaper Plots

Time to convert 10 years of CESM data from time slice to time series.
Low Resolution Timings
Original method vs. Swift vs. pyAverager

<table>
<thead>
<tr>
<th></th>
<th>CAM FV</th>
<th>CAM SE</th>
<th>CICE</th>
<th>CLM</th>
<th>POP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCO</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>SWIFT</td>
<td>5</td>
<td>5</td>
<td>0.4</td>
<td>1.2</td>
<td>7</td>
</tr>
<tr>
<td>pyAve</td>
<td>4</td>
<td>3</td>
<td>0.2</td>
<td>1.5</td>
<td>3</td>
</tr>
</tbody>
</table>

High Resolution Timings
Original method vs. Swift vs. pyAverager

<table>
<thead>
<tr>
<th></th>
<th>CICE</th>
<th>CAM</th>
<th>CLM</th>
<th>POP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCO</td>
<td>27</td>
<td>215</td>
<td>14</td>
<td>306</td>
</tr>
<tr>
<td>SWIFT</td>
<td>6</td>
<td>102</td>
<td>7</td>
<td>92</td>
</tr>
<tr>
<td>pyAve</td>
<td>2</td>
<td>16</td>
<td>4</td>
<td>21</td>
</tr>
</tbody>
</table>

Downloads Available

PyReshaper v0.9.1
PyAverager v0.1.0
PyTools v0.1.0

Required Packages

NumPy
mpi4py
PyNIO
Contact Info
mickelso@ucar.edu