IOAPITOOLS: A PYTHON ANALYSIS ENVIRONMENT FOR CMAQ, SMOKE AND MCIP

Alexis Zubrow*
CISES, University of Chicago, Chicago, IL USA

This talk presents ioapiTools, an interpretive software framework for manipulating and analyzing IOAPI formatted data. The ioapiTools software leverages the flexibility and power of python to create a comprehensive analysis environment or framework, which provides simple functions for the input, output, manipulation, and display of IOAPI data and metadata. At the same time, ioapiTools can be easily adapted to meet the analysis needs of individual researchers.

Air quality models require the input and output of a large amounts of data. The CMAQ, SMOKE, and MCIP models use a unique data format called IOAPI, an adaptation of the more common netCDF data format found in many scientific fields. The profusion of software tools developed for netCDF files facilitates the reformatting, manipulating, and plotting of netCDF data. Unfortunately, these tools allow for only a limited access to the IOAPI metadata. In particular, they have difficulty writing IOAPI formatted files or assisting the researcher's physical intuition in their analysis. The primary tool to directly and fully access IOAPI data, the IOAPI library, requires one to compile code. Although compiled code is very efficient, it does not promote interactive analysis with one's data.

The ioapiTools was developed in the python language to address the above needs. Why python? First, python is an interpretive language. A researcher simply uses a python shell to interactively run commands, similarly to commercial packages such as Matlab or IDL, or she can write a non-compiled, python script. Second, python is a relatively easy software language to learn and has a high degree of clarity. Subsequently, scientists can easily develop their own scripts and functions and importantly, can understand those software fragments months or years later. Because of these two strengths, the software development and debugging process is much more rapid than in compiled languages. Third, python is an excellent glue between different software languages, which expedited my wrapping of the Fortran IOAPI library (i.e. creating a python interface to the low level library). Fourth, python is extendible. A scientist can take advantage of a growing list of modules (basically a python library) that gives access to new functions or capabilities. The growing popularity of python in the scientific community means that there are a large number of numerical, scientific, and graphical packages to add to the basic python language.

In addition to the general strength of python, the ioapiTools module provides specific functionality for CMAQ modelers. First, the ioapiTools can both read and write IOAPI formatted files. All of the metadata is accessible and modifiable through the module's interface. Second, not only can researchers write out IOAPI formatted files, but they can also write CF netCDF (Climate and Forecast compliant netCDF), ASCII, or binary files. If researchers export to a CF netCDF file, they can use a much larger suite of software tools to analyze the data. Third, plotting capabilities are built directly into the ioapiTools, including contour, vector, and contour overlay plots. Fourth, ioapiTools includes coordinate transformations. A scientist's physical intuition can be used to spatially or temporally subset or query the data sets. For example, she can select geographic regions based on their latitude and longitude, or a temporal range based on dates. Fifth, the ioapiTools are built on top of cdat (Climate Data Analysis Tool). Over the last couple of years, there has been a convergence between the climate and air quality communities. By building upon cdat, the ioapiTools helps act as a conceptual bridge between these two communities.

Presently, the ioapiTools is being distributed under an open source license. The module is stable, but is being constantly developed and expanded. As the user base increases, I expect that the ioapiTools will continue to increase its breadth of functions.

* *Corresponding author*: Alexis Zubrow, Center for Integrating Statistical and Environmental Science, University of Chicago, 5734 S. Ellis Ave, Chicago, IL 60637; email: azubrow@uchicago.edu