An Object-oriented Python Implementation of an Intermediate-level Atmospheric Model

Johnny Wei-Bing Lin
Physics Department, North Park University, Chicago, Ill.

Features
The qtcm package (Lin 2008) is a Python wrapping of the Neelin-Zeng (2000) Quasi-equilibrium Tropical Circulation Model, a primitive equation-based intermediate-level atmospheric model written in Fortran. By using Python as a wrapper, we create an integrated modeling and visualization environment with the following features:

Interactive model runs as Python objects: Visualization and analysis is integrated in with model execution.

Execution control using run lists: Enables runtime control of subroutines execution order and content.

Doing science more easily: In this interactive modeling environment, the traditional sequence of “hypothesis → model → visualization and analysis” is made nonlinear and flexible, enabling more science questions to be easily addressed.

Interactive modeling with qtcm
Fig. 1 shows a screenshot of an interactive Python session running an instance of the qtcm tropical atmosphere model (QTCM). During a model run, you have access to all model variables. The visualization is done interactively at runtime. The screenshot also shows how you can change model variable values with an assignment statement and continue the model run by calling the run_session method again.

Execution control using run lists
Because Python is an interpreted language, subroutine execution order and content is not fixed during runtime. QTCM uses “run lists,” lists of string names, to describe what subroutines are called the Python method to generate the two plots. The third line shows variable substitution for prognostic variable u1 (doubling the existing value), and the fourth line will run the model for another 50 days when executed.

The 180 day model run took a little over a minute of wall-clock time on a 1.83 GHz Intel Core Duo with 1 GB 667 MHz DDR2 SDRAM running Mac OS X version 10.4.11. The horizontal grid for the model is 5,625 × 3,75 degrees longitude and latitude.

Doing science more easily
Because the object-oriented Python wrapper provides so much flexibility at run time, QTCM gives the opportunity to automate more of the steps involved when using models to answer science questions (Fig. 2). For instance, a conditional test of a model’s solution space, instead of requiring multiple versions of source code, makefiles, and shell scripts, can be coded as a simple if-then loop, something like this:

```
model = Qtcm(**inputs)
while <condition true>
    <alter prev snapshot depending on condition>
    model.sync_set_py_to_snapshot(snapshot=prev)
    model.run_session()
    prev = model.snapshot
```

References and acknowledgments
Lin, J.-W.-B., 2008: qtcm 0.1.2: A Python implementation of the Neelin-Zeng Quasi-Equilibrium Tropical Circulation Model, Geosci. Model Dev. Discuss., 1, 335–368.

For more information
Please contact Johnny Lin at:
Email: jlin@northpark.edu
Personal home: http://www.johnny-lin.com
qtcm home: http://www.johnny-lin.com/py_plgs/qtcm
Paper: http://www.geosci-model-dev-discuss.net/1/335/2008