A Python Implementation of an Intermediate-level Tropical Circulation Model and Implications for How Modeling Science is Done

Johnny Wei-Bing Lin

Computing and Software Systems Division, School of STEM, University of Washington Bothell
Physics and Engineering Department, North Park University, Chicago, Ill.

Features

The `qtcm` package (Lin 2009) 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 subroutine execution order and content.

Doing science more easily: In this interactive modeling environment, the traditional sequence of "hypothesis ➜ modeling ➜ 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 (`model`). During a model run, you have access to and can change all model variables. The visualization is done interactively at run time.

Execution control using run lists

`qtcm` uses "run lists," lists of string names and dictionaries, to describe what subroutines are executed and in what order. For instance:

```
>>> model = Qtcm(compiled_form='parts')
>>> print model.runlists['qtcminit']
['__qtcm.wrapcall.wparinit',
'__qtcm.wrapcall.wbndinit', 'varinit',
{'__qtcm.wrapcall.wtimemanager': [1,]},
'atm_physics1']
```

Because lists can be manipulated at run time, subroutine execution content and order is completely changeable at run time.

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 `while` 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
```

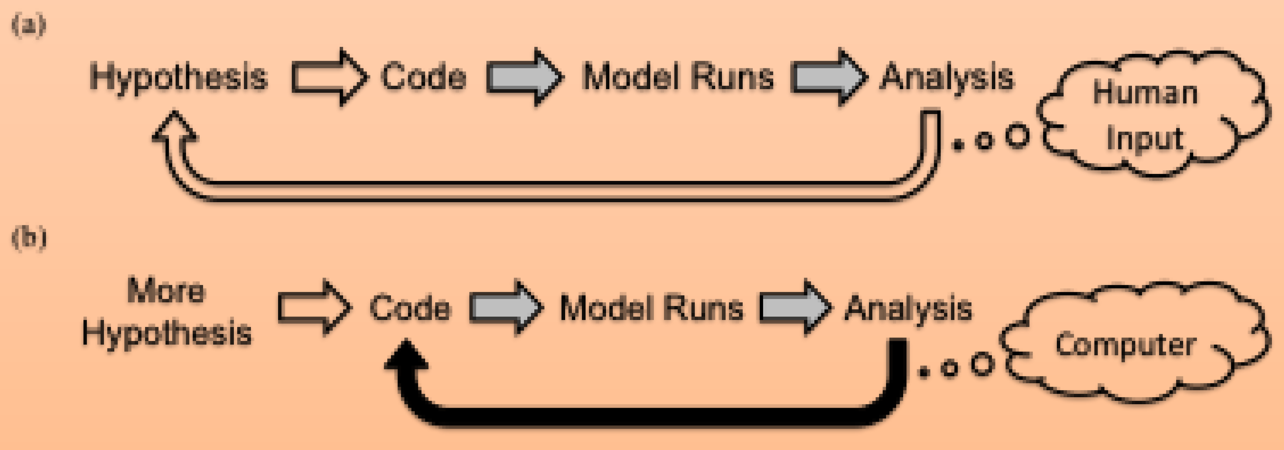

Fig. 2. (a) Traditional and (b) revised sequence of using climate models to address science questions (Lin 2009).

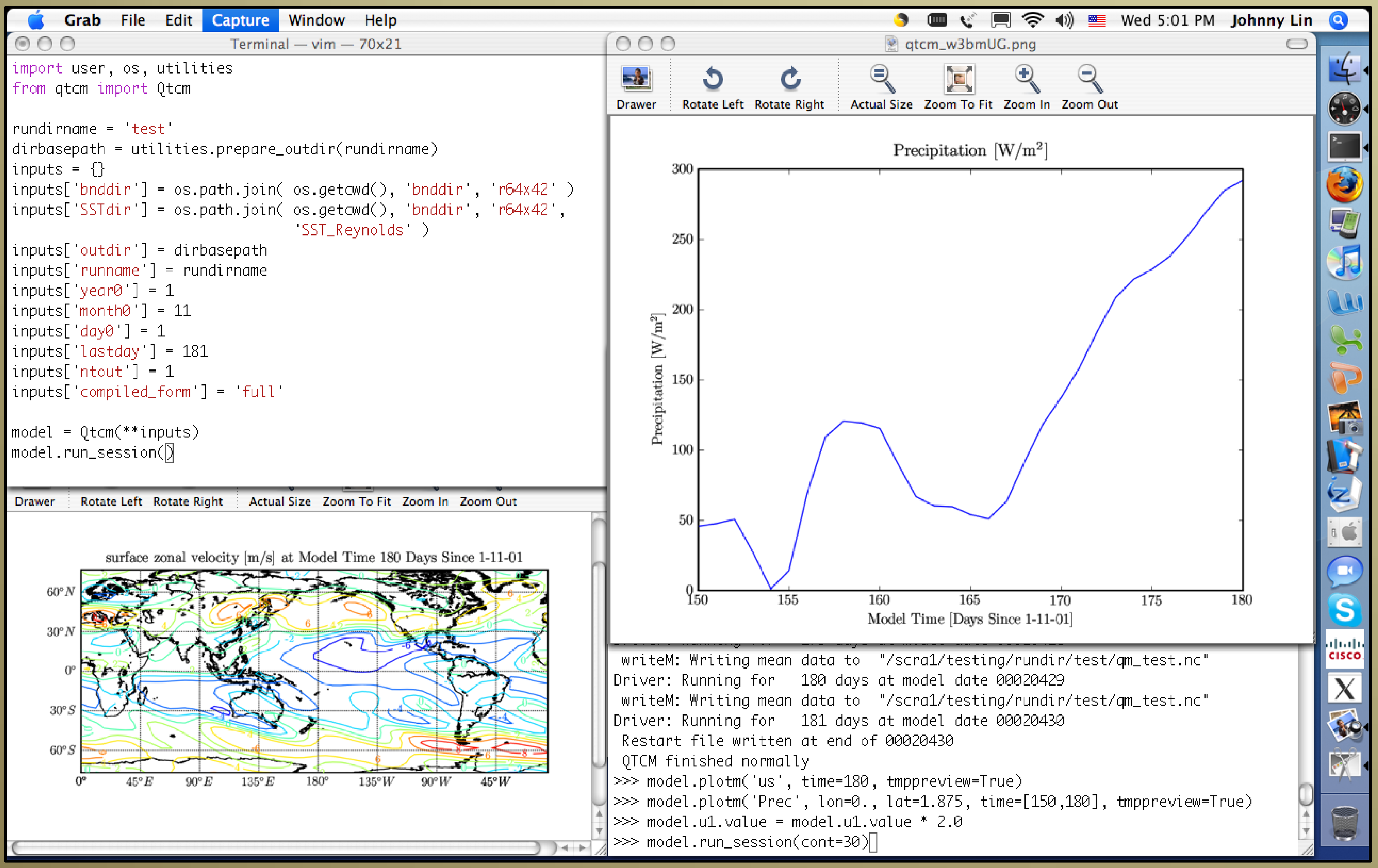

Fig. 1. Screenshot of an interactive integration of a `qtcm` model instance.

The upper-left window shows the code for initializing the model instance and running 180 days of simulation.

The lower-right window shows the run session. The first two lines in the window called the `plotm` 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 30 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.

For more information

Email: jwblin@uw.edu
Personal home: http://www.johnny-lin.com
`qtcm` home: http://www.johnny-lin.com/py_pkgs/qtcm
Paper: http://www.geosci-model-dev.net/2/1/2009/gmd-2-1-2009.html

References and acknowledgments

Lin, J. W.-B., 2009: qtcm 0.1.2: A Python implementation of the Neelin-Zeng Quasi-Equilibrium Tropical Circulation Model, *Geosci. Model Dev.,* 2, 1–11.

Neelin, J. D. and N. Zeng, 2000: A quasi-equilibrium tropical circulation model—formulation, *J. Atmos. Sci.,* 57(11):1741–1766.

Thanks to David Neelin, Ning Zeng, Matthias Munnich, Alexis Zubrow, Christian Dieterich, Rodrigo Caballero, Michael Tobis, and Ray Pierrehumbert. Thanks to God for allowing the `qtcm` package to run. Early development of `qtcm` precursors was carried out at the University of Chicago Climate Systems Center, funded by the National Science Foundation (NSF) Information Technology Research Program under grant ATM-0121028. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the NSF. Nearly all of the contents of this poster was previously presented at other meetings.