A simple object-oriented framework for making sea-ice models “plug-and-play”

Johnny Wei-Bing Lin
Computation Institute, University of Chicago

Introduction

Historically, sea-ice models have been developed incrementally and in compiled languages like Fortran. This has limited their modularity and reusability. The code is difficult to maintain and keeps it from becoming brittle.

Fig. 1 shows a snippet of Fortran code from an ice model. This snippet calls subroutine that calculates the sea-ice budget. The argument lists are long and unwieldy, the variables contain no metadata (and thus are not easily propagated), and the grid is fixed, making it hard to couple the model to other climate models or components.

Compiled languages such as Fortran are also not interactive.

Modern computer languages have structures that overcome many of these difficulties. Here we implement the Semtner (1976) “u” and “n”-layer thermodynamic models in the open-source language Python as an object-oriented framework that results in a source code that is more “plug-and-play” than existing models. Because Python has a robust standard library with interactive code, the framework described here can be extended in a straightforward way to “wrap” existing Fortran sea-ice models such as CISM.

Modular sea-ice models

Other modular languages usually lack tools to manage the variable namespace. One result is that subroutine argument lists are “hard-wired” and lengthy. Modern languages avoid this through dictionaries and the creation of “domain” data structures that allow the right variables to be available and used when needed.

Fig. 2a shows code used to initialize the “u”-layer model (for clarity, here to initialize only one item not shown). The parameters, boundary conditions, and variables, along with their metadata, are stored in the _init_state object. When the model is initialized (last line), only one argument is then needed. Note that the _init_state object is a Fortran common block. Because it also contains variable metadata, this object is very capable.

Fig. 2b shows initialization code for the “n”-layer Semtner model. The code is advanced to 0.0, except for the last line. What this illustrates is that the initial conditions are the same, since they cannot be the ice layer temperature variable in the _init_state run.

Interactive modeling

Fig. 3 shows a screenshot of an interactive Python session running the “u”-layer Semtner (1976) model in some configuration mode. Because Python is an interpreted language, there is no separate compilation cycle, and an interactive session you have access to all variables in the current scope. The upper left-hand window shows some source code and the lower right-hand window shows the result as light lines between two years of integration.

The visualization was done interactively at run-time. A user can also change parameter values at run-time and continue integration using these new variables. This makes climate scientists to use the sea-ice models interactively.

Grid independent models

Because object-oriented languages allow you to carry within object all the metadata needed to operate on the variable data also in that object, programs written in the object-oriented framework can readily be made grid independent.

Fig. 4 shows the code used to calculate change in snow thickness due to melting in the “n”-layer model. The first three terms on the right-hand side are merely omitting only places with ice and snow and are shown or equal to the melting point have snow melting.

The magnitude of the change is the timestep times the net heat flux divided by the heat of fusion of snow.

Conclusions

Modern object-oriented computer languages enable us to create sea-ice modeling frameworks that are modular, interactive, and grid independent. Such frameworks can also be used to model and other climate model components. Utilizing frameworks like these promises to enable climate scientists to focus less on programming and more on using the models for hypothesis testing and physical insight.

Reference and acknowledgments

This work is supported in part by the National Science Foundation and the U.S. Department of Energy’s Office of Science. We gratefully acknowledge the support of the U.S. Department of Energy’s Office of Science.

For further information

http://climate.uchicago.edu/