Computers & Geosciences 90 (2016) 152–161

Contents lists available at ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

Case study

Towards uncertainty quantification and parameter estimation for Earth system models in a component-based modeling framework

Scott D. Peckham [a,*], Anna Kelbert [b], Mary C. Hill [c], Eric W.H. Hutton [a]

[a] INSTAAR, University of Colorado, Boulder, CO 80309, United States
[b] CEOAS, Oregon State University, Corvallis, OR 97331, United States
[c] Department of Geology, University of Kansas, Lawrence, KS 66045, United States

ARTICLE INFO

Article history:
Received 20 January 2015
Received in revised form
2 March 2016
Accepted 3 March 2016
Available online 4 March 2016

Keywords:
Model uncertainty
Modeling frameworks
Component-based modeling
Optimization
Inverse problems
Nonlinear least squares
Parameter estimation
Longitudinal river elevation profiles

ABSTRACT

Component-based modeling frameworks make it easier for users to access, configure, couple, run and test numerical models. However, they do not typically provide tools for uncertainty quantification or data-based model verification and calibration. To better address these important issues, modeling frameworks should be integrated with existing, general-purpose toolkits for optimization, parameter estimation and uncertainty quantification.

This paper identifies and then examines the key issues that must be addressed in order to make a component-based modeling framework interoperable with general-purpose packages for model analysis. As a motivating example, one of these packages, DAKOTA, is applied to a representative but nontrivial surface process problem of comparing two models for the longitudinal elevation profile of a river to observational data. Results from a new mathematical analysis of the resulting nonlinear least squares problem are given and then compared to results from several different optimization algorithms in DAKOTA.

1. Introduction

Many Earth science domains rely on numerical modeling to gain a better understanding of Earth system processes. Modeling addresses a wide variety of problems in the realms of climate, weather, hydrology, land surface dynamics, geodynamics, geophysics, hydrogeophysics and structural geology, among others. Earth system models are based on physical, chemical, biological and stochastic processes that make it theoretically possible to predict changes likely to occur at, below, or above a particular location on Earth in response to various types of forcing. Data-based model verification and validation – including more formal data integration through model parameter estimation – and quantification of ever-present uncertainty are critical in order to develop reliable numerical models for observed Earth processes.

The Community Surface Dynamics Modeling System, or CSDMS, is one example of a component-based modeling framework (Peckham et al., 2013; Syvitski et al., 2014), employed in the realm of Earth surface process dynamics, with capabilities currently being extended to deep Earth process modeling. Just as CSDMS provides interoperability and coupling mechanisms for process-based models, it could also provide simplified access to model analysis programs. In this paper, we discuss extensions to CSDMS that would be required for its component-based framework to interoperate with uncertainty quantification and parameter estimation (inverse modeling) toolkits.

2. Background: models and modeling frameworks

2.1. What is a model?

There are many possible definitions of the word *model*. This paper is concerned with *computational models* that predict the evolution of one or more system state variables over time as a function of observations at a given start time. These predictions are made using a set of equations that express laws of physics and other constraints on the problem of interest. Laws of physics are often expressed as differential equations that include a time derivative, and computational models use a discretization of space and time and some combination of *numerical methods* to solve these governing equations. Models generally require values for one or more *input variables*, often used to describe the initial state of the system and often specified as spatial scalar or vector fields. These may be measured or estimated and are distinct from the model's *design parameters* (also called control, model or

* Corresponding author.
E-mail address: Scott.Peckham@colorado.edu (S.D. Peckham).

http://dx.doi.org/10.1016/j.cageo.2016.03.005
0098-3004/

configuration parameters), that must be specified in the equations that define the model itself. A model run generates numerical values for *output variables* (i.e. simulated observations or predictions) that can be compared to observations. A very simple example is given by $y = c\,x^p$, where x and y are input and output variables, respectively, and c and p are design parameters.

2.2. What is a modeling framework?

Over the last decade, a number of different *modeling frameworks* have emerged, both within academia and at several different federal agencies. An example from the academic modeling community is the NSF-funded CSDMS project (cited in the Introduction) which primarily serves the Earth surface process modeling community. Other examples from the federal or operational modeling community include

- ESMF (Earth System Modeling Framework), which primarily serves the atmosphere and ocean modeling community,
- OMS (Object Modeling System), developed by the USDA (US Department of Agriculture) primarily for agricultural modeling and
- FRAMES (Framework for Risk Analysis in Multimedia Environmental Systems), developed by the US EPA (Environmental Protection Agency), primarily for environmental modeling.

(Hill et al., 2004; David et al., 2002; Whelan et al., 1997). A project called Earth System Bridge, funded as a building block in NSF's EarthCube initiative, is developing adapters that make it easy for any given model to be prepared as a plug-and-play component that can be used in (or moved between) multiple modeling frameworks, including those above.

The intent of all such modeling frameworks is to provide a software environment in which users can choose models from a collection and easily couple them to create customized, composite models in a plug-and-play manner. This facilitates code re-use and interoperability. The models in the collection may span a wide variety of different physical processes and are often written by many different authors, typically experts in their field. In many cases, the input variables required by one model can be provided by another model in the collection, so there is strong motivation to couple them. However, the models typically differ in many ways, such as their programming language, computational grid, time-stepping scheme, variable names and units. In addition to providing a simple mechanism for coupling models, modeling frameworks typically contain *service components* or *mediators* that automatically reconcile differences between the models that would otherwise prevent them from sharing variables. Examples of mediators include spatial regridders, time interpolators, unit converters and semantic mediators. These mediators and other capabilities of the framework – such as the ability to write composite model output to different file formats with standardized metadata, or to provide a graphical user interface (GUI) and help system – provide both model users and developers with significant added value.

There is a strong interest in adding a new capability to modeling frameworks, namely the ability to track and analyze uncertainty either for a single (stand-alone) model or for a coupled set of models. For example, this is one of the major goals of the second funding cycle of the CSDMS project. Since several powerful, integrated packages for uncertainty analysis already exist (Section 4), integrating one or more of them into a modeling framework seems like the best way to achieve this goal. One such package, called DAKOTA (Adams et al., 2013b, 2013a) is of particular interest because it provides a unified interface to a large collection of open-source packages for optimization and uncertainty quantification.

DAKOTA and similar packages offer an impressive suite of uncertainty analysis tools, including tools for model sensitivity analysis (e.g. sampling methods to explore the design parameter space) as well as inverse modeling (or parameter estimation). However, the sensitivity analysis tools are easier to integrate within a modeling framework because they do not usually require capabilities beyond what a typical model (or composite model) already provides. By contrast, inverse modeling requires construction of a suitable objective function and computation of derivatives and is also affected by how models are coupled. So although we are interested in bringing all of the capabilities of packages like DAKOTA into modeling frameworks like CSDMS, this paper will focus on what a modeling framework must do to accommodate inverse modeling. To set the stage, the next section provides a very brief, self-contained overview of inverse modeling. For a more extensive treatment, see Tarantola (2005), Caers (2011) or Aster et al. (2013).

3. Background: inverse modeling methods

Forward modeling simply refers to running a computational model for a *given* set of input variables and design parameters to generate output variables. *Inverse modeling* refers to efforts to invert this process, that is, to determine what a model's input variables and/or design parameters would need to be set to in order to generate a *given* set of output variables. In most cases, this inverse problem is *ill-posed*, meaning that there is not a unique set of input variables and design parameters that can produce a given output, but rather multiple sets. However, *regularization methods* can be used to introduce additional criteria that discriminate between and preferentially select from these multiple sets.

A forward model's performance can be quantified by defining a metric that measures the “distance” between its output variables (or predictions or simulated data) and independent measurements (or observations). Differences between corresponding observed and predicted values are known as *residuals*, and this metric – known as the *penalty, cost or objective function* (Section 3.1) – is typically a function of the residuals, input variables and design parameters. Inverse modeling is concerned with how to make forward models perform as well as possible (model calibration), or with seeking the optimal input variables to predict observations to within measurement error. They therefore make use of *optimization methods* that seek to minimize an objective function, often subject to additional constraints.

Earth system modelers range widely in their familiarity with and adoption of inverse modeling methodology. For example, groundwater modelers have a long history of using inverse models, while sediment transport modelers do not. Inclusion of these methods in modeling frameworks should encourage broader use of these methods.

3.1. Constructing an objective function

The *objective function* must be a *metric* that measures a forward model's performance, or the abstract “distance” between observed and model-predicted or simulated values. There are many different metrics that can be used, such as those based on the one-parameter family of L^p norms, given by

$$\| \mathbf{y}^{(obs)} - \mathbf{y}^{(sim)} \| = \left(\sum_{k=1}^{n} \left| y_k^{(obs)} - y_k^{(sim)} \right|^p \right)^{1/p} \tag{1}$$

where $\mathbf{y}$ is a vector with components y_k and $p > 0$ is a scalar. The case where $p=2$, or the L^2 norm, is the basis of the popular *least squares* metric. While this metric gives disproportionate weight to