Towards a Shared Modeling Terminology and Problem Specification Framework

Deborah Khider, USC/ISI
Yolanda Gil, USC/ISI
Daniel Garijo, USC/ISI
Kelly Cobourn, VT
Chris Duffy, PSU
Armen Kemanian, PSU
Scott Peckham, CU
Ben Watkins, Kimetrica
Alison Campion, Kimetrica
Chris Preager, Kimetrica
Suzanne Pierce, UT Austin
Daniel Hardesty Lewis, UT Austin
Anna Dabrowski, UT Austin
Cheryl Porter, University of Florida
Marty Landsfeld, UC Santa Barbara
Michael Puma, Columbia University
Bernhard Schauberger, Potsdam Institute for Climate Impact Research (PIK)
Amy Sliva, CRA
Clayton Morrison, University of Arizona

This document proposes an overarching modeling problem framework and definitions for terms used in the World Modelers program in TA2-TA3 modeling, also called bottom-up modeling. There are several reasons to propose a shared conceptual framework and terminology. First, there is a high degree of ambiguity in the use of terms such as “scenario”, “model”, “intervention”, “data”, etc. A precise characterization of these terms in a computational framework will reduce that ambiguity. Second, a precise characterization of terms can create a shared understanding among TA2-TA3 groups and facilitate communication and integration with TA1 groups. Finally, the use of a shared terminology will also facilitate user training and therefore program-wide evaluations.

Table of Contents

Basic Definitions for Quantitative Modeling

This document defines basic terms in World Modelers bottom-up modeling, i.e., using models developed based on data about past observations that attempt to predict future observations.

A *complex dynamical system* can be characterized as a set of interdependent *variables* (input, output, endogenous, exogenous) that describe the state of the system. *Observations* may be collected for some of its variables if they are accessible and measurable.

Evaluation of models can be carried out by comparing the predictions that they generate with actual observations. *Uncertainty* and *model sensitivity* can generally be studied by comparing the results of many model runs under different conditions, although both uncertainty and sensitivity are multifaceted and nuanced.

When two or more models are integrated, there are many ancillary steps involving data preparations (e.g., normalization, cleaning) and data transformations (e.g., reformatting) that can be very hard to manage. In order to assist users to create integrated models, a computational framework is needed to capture such complex processes and to define in precise terms different modeling aspects.

We begin this document by casting integrated modeling in terms of computational workflows (sometimes called pipelines), workflow executions, and workflow ensembles. We then describe different kinds of modeling questions and define precisely how they can be realized in this computational framework.

Types of Models

We can distinguish several types of models:

- *Theory-based models* based on theoretical foundations of the phenomena governing the system, such as physical laws, biological processes, or economic theories. They are sometimes called *mechanistic models* because the theory explains the causality and mechanisms behind the model.
- *Empirical models* based on patterns that emerge among the observed variables, often derived through statistics (*statistical models*) or through machine learning.
- *Expert-specified models*, where the model is designed by hand based on expert knowledge. This is a common type of model in social sciences, where an expert may express what variables are relevant and give some general information about how they are related.
- *Informed models*, which are generated automatically through text extraction from a collection of documents, and express what variables and entities are mentioned and link them based on what the text indicates.

- *Hybrid models*, which combine one or more types of the above types of models

In World Modelers, bottom-up models could be characterized as theory-based models, empirical models, expert-guided models, or hybrids of those. They are models developed based on past observations. Top-down models could be described as informed models.

Workflows, Realizations, Ensembles

Computational Workflows and Workflow Components

A *computational workflow* describes a set of computations as a graph. The computations are the nodes, and the links reflect their data dependencies. Workflows are sometimes called pipelines, which evoke how commands could be piped in a sequence in Unix, but the dataflow of a workflow is rarely sequential.

A workflow is often expressed as a directed acyclic graph (DAG), though this is not very conducive to representing temporal dependencies among variables, or coupled modeling, or feedback loops. Reproducibility and provenance are important, so tracking what was executed is a priority and is harder to do with more complex workflow frameworks.

Each computation in a workflow is called a *workflow step*, *workflow component*, or *workflow task*. A workflow component is described by the specific software that is executed, the *signature* of the invocation (i.e., the invocation command), and the selected *input datasets* (or *input data*) and *input parameters*. Upon execution, a workflow component generates *output datasets*.

From a modeling perspective, input data and parameters capture *modeling variables* that are used to specify the *initial situation* (or initial state). This may include *raw data* (e.g., direct observations collected through an instrument) or *processed data* (e.g., raw data that has been cleaned and reformatted). The output datasets of models would contain values for the *response variables* of interest.

Model outputs are not only variables of interest, they may include diagnostics, statistics, logs, and other ancillary kinds of data.

A given input dataset or output dataset may contain several variables (e.g., a dataset may be a table where the first column is a timestamp and each column is a variable). A dataset can be implemented in a variety of formats (a file vs a collection of files, a shapefile vs NetCDF), and their documentation typically describes how the variables are represented within the dataset.

Computationally, the difference between input data and input parameters is that input parameters are used to modulate the behavior of the software component while input data

reflect the initial state of the system before the model is run. In modeling, those differences may not be so crisp, since variables can be represented in input data or input parameters.

User-defined parameters are input parameters that the user provides for a specific execution. It is important for the system to track these in order to provide accurate provenance of how execution results are generated.

A *workflow template* specifies the dataflow among the computational steps but without specifying any particular input data and parameters. An *execution-ready workflow* specifies all input data and parameters. Technically, an *execution-ready* workflow specifies the *bindings* for all input data and all input parameters of a workflow template. A binding is a mapping between an input file and how that file is used in the workflow; same for a parameter. The *execution-ready* workflow may also have additional steps to properly manage the workflow execution (e.g., data movement, storage management, etc.).

A *high-level workflow template* is a very abstract description of a realization, that is, a workflow template that may be missing steps. For example, it may include steps for the models to be run but no data transformation steps.

Workflow Components for Models and Data Transformations

A workflow component can be any software, and therefore a workflow component can be:

- Theory-based models, predictive models that go from initial conditions to final conditions through time steps
    - A physics-based simulation model
        - E.g., a hydrology model
    - A biophysical simulation model
        - E.g., an agriculture model
    - An agent-based model based on networked behaviors
    - Economic models, social models, conflict models
    - A simple model with an equation that expresses how to generate a variable of interest from a set of other variables whose values are known,
        - E.g.: relative humidity can be calculated from temperature and dew point temperature
- An empirical model derived from data through machine learning or statistics
    - E.g., a model of household poverty level
- A data transformation that reformats, converts, regrids, or otherwise manipulates some given data
    - E.g., converting a shapefile into NetCDF
- Data extraction from unstructured documents and tables
- A visualization step
- A data comparison step
- A model comparison step

- A subworkflow composed of steps

Complex off-the-shelf models are often complex software packages that bundle together a lot of different functionalities that may be modular and could be selected and combined. When specific components of a model software package are selected (e.g., specific processes to simulate or specific mathematical functions for those processes), that constitutes a *model configuration*. That is, a model configuration is a particular setting of the modeling software that performs a unique function expressed with the invocation command as well as input data and parameters.

A *calibrated model* is a set of input datasets and input parameters for a model that result from a calibration process that runs the model and compares it to historical observations in order to adjust it to a specific physical environment or other conditions.

Model spin up conditions is a subset of input data and input parameters that are used to achieve model equilibrium conditions that are used to run the model for the subsequent time period of interest together with additional input *forcing conditions* (also called *stressing conditions*) to initialize the model for that period. Identifying model spin up conditions is useful because they cover a time period that is not necessarily of interest to the analyst, so it should not be included in the results.

Realizations

A *realization* is a single run of an integrated model, where given a fixed input situation and parameter settings (i.e., given input datasets and parameters), the integrated model generates a prediction (i.e., output datasets). A realization is a workflow execution.

A *realization specification* consists of an input situation (input data and input parameter settings) together with the model(s) form which can then be submitted for execution in order to create a realization. A realization specification is an execution-ready workflow.

An *incomplete realization* is a realization specification that failed to execute, for example because a model in the workflow does not converge or because of a bug in some component.

An *incongruent realization* is a realization with a complete execution that is considered erroneous because the predictions are not consistent with observations, physical laws, or known system behaviors.

For many models, given fixed input datasets and parameters the prediction is always the same since they are *deterministic models*. This is not the case for all models. In non-deterministic models, it is possible that a first execution fails while the second one succeeds (eg if the second one uses a different seed).

An *iterative realization* is an nth execution that occurs in the context of a Monte Carlo or similar approach of n-iterations. For example, we may run multiple iterations of a workflow and test the sensitivity of target variables to changes in upstream parameters.

Reference Realizations

A *reference realization* is an execution of an integrated model that is compared against one or several others. This is often referred to as a *baseline*. For example, a reference realization can be the prediction using similar precipitation levels as the previous year in order to compare with realizations with higher precipitation levels and others with lower precipitation levels.

Workflow Ensembles

A *workflow ensemble* is a collection of realizations that have a common theme. The theme can be implemented as a variation over some workflow elements while the rest of the workflow is considered fixed. For example, an ensemble may vary the value of an input parameter while all else (e.g., the input data, the models) remains the same.

In some cases, the workflow elements that are not in the theme can be considered fixed and will be the same for all the ensemble elements. This is the case when a parameter is varied, since the data preparation and modeling steps of the workflow will typically be exactly the same. Note that the workflow elements not in the theme can be considered fixed but in reality they will not be the same. For example, if a variation in the initial data is the theme it may not be possible to use the same model configuration for all the initial conditions, and if the model configuration varies then the data preparation steps may also vary.

Nested ensembles have multiple levels of themes within themes, where each element of an ensemble can be another ensemble. For example, a parameter p1 may be varied in an ensemble, with each of the elements has a fixed value of p1 but is an ensemble where p2 is varied.

Workflow Ensemble Types

There are several types of workflow ensembles, defined by the theme:

- *Workflow parameter ensembles*: a parameter value varies. It is composed of iterative realizations.
- *Workflow data ensembles*: an input dataset is obtained from different data sources
- *Workflow model ensembles*: a modeling step is implemented with alternative models.

Workflow ensembles are useful for sensitivity analysis and estimating overall model uncertainty.

Workflow Ensemble Specification, Construction, Execution, and Aggregation

A *workflow ensemble specification* includes:

1. the theme of the ensemble
2. a comparable structure for the remainder of the realizations
3. (optionally) a reference realization or baseline

Workflow ensemble construction involves generating a realization for each element of the ensemble following the specification.

Workflow ensemble execution involves running the realization for each ensemble element.

Workflow ensemble aggregation involves comparing or combining the results of the executions of the workflows for the ensemble elements. The reference realization, if specified, can be used as a baseline or fixed point for comparing other realizations in the ensemble. The ensemble elements can also be aggregated to estimate uncertainty.

Machine Learning Ensembles

A *machine learning ensemble* consists of several models learned from the same training data whose results are combined using different strategies, and are very popular for developing empirical models. The models usually have the same base learning algorithm (e.g., a random decision tree algorithm), and each element of the ensemble is learned with a different parameterization. The results of all the elements in the ensemble can be combined by weighted voting, voting with equal weight, Bayesian combination, etc.

A machine learning ensemble can be a workflow model ensemble, a workflow parameter ensemble, or a nested ensemble. A machine learning ensemble could be implemented as a workflow ensemble that runs each of the elements, and includes a combination step that would integrate the results of all the workflows. In this sense, a machine learning ensemble spans the ensemble construction, execution, and aggregation aspects described above.

Diagnostic, Prognostic, and Counterfactual Modeling Questions

Prognostic Questions

A *prognostic question* uses *forecasts* (i.e., data about possible initial conditions or situations) in order to understand how some set of variables of interest will behave in the future. Prognostic questions may look into a few days or into hundreds of years into the future.

Examples of prognostic questions are: Can we expect flooding in South Sudan in the upcoming rainy season? How does a 2-sigma drought affect water availability for crops?
How will climate change impact water resources (e.g. reservoirs in South Sudan) by the year 2030?

Formally, a *forecast specification* consists of all the input data and parameters needed by the models in order to carry out one realization. For weather data, the data would be provided by forecast systems, which can provide weather outlook up to a year in advance. Medium-range forecast (10-day forecast) are useful for cyclones and extreme weather conditions. Extended range forecast (46-days) focus mainly on the week-to-week changes in weather, including potential tropical cyclone activity. Long-range forecast, which can be up to a year in advance, rely on aspects of Earth system variability which have long time scales (months to years) and are, to a certain extent, predictable. The most important of these is the El Niño-Southern Oscillation cycle.

Diagnostic Questions

A *diagnostic question* uses data about past situations in order to understand how some set of variables of interest behave. Data about past situations may be available from observations, but when variables cannot be observed the data may be *hindcasts* of those variables, i.e., their probable values. Diagnostic questions are useful to identify the source of a problem that has already occurred, and to understand how to avoid the same problem in the future.

An example of a diagnostic question is: “What caused the crops to fail in 2017 in South Sudan?”

Formally, a *hindcast specification* consists of all the input data and parameters needed by the models in order to create a realization specification that when executed can be a realization that corresponds to the hindcast. A hindcast specification may include observed values or probable values.

In the case of weather data, hindcast specification will mostly be obtained from *reanalysis* (or *reforecasting*) and merged satellite/gauge products. A climate reanalysis gives a numerical description of the recent climate, produced by combining models with observations. The estimates are produced for all locations on earth and they span a long time period that can extend back by decades or more. Reanalysis products are often derived from data assimilation and forecast systems to reanalyze archived observations.

Counterfactual Questions

A *counterfactual question* (often asked as a "what if" question) uses data about a situation to explore alternative timelines that are either diagnostic or prognostic in nature.

Examples of counterfactual questions are: What if there is a drought during the next planting season? What if roads had not been inundated during the last flood?

Potential human interventions (see below) can be analyzed by setting up counterfactual questions. Counterfactuals questions are also used to study natural forces (e.g., storms, earthquakes) that are not controlled by humans, and would be exogenous events that affect the system being modeled.

Interventions are not always part of a counterfactual question. For example, different precipitation increases (20%, 30%) would be posed as different counterfactual questions, and none of those would be considered an intervention or an exogenous event.

Counterfactual questions often require the use of a *baseline* (or *status quo*) for comparison. This baseline is a reference realization.

Indicators and indexes

An *indicator* is a variable that is identified as playing a special role, namely to help characterize a complex property of a system being modeled. Indicators are often mentioned in modeling questions. They are often chosen because they are meaningful to end users. A *proxy indicator* is an indirect way to measure variables of interest.

An *index* is a combination of 2 or more variables (aka indicators) that can be measured (any of which could be seen as indicators, consistent with above) with the goal of using a single number for assessment and comparison purposes. indexes are designed to give a good aggregate value of relevant variables to understand a situation.

indexes may be used as part of any question instead of or in addition to response variables. For instance, a drought index can indicate the severity of dry conditions, which depends on precipitation and temperature (For instance, see the Palmer Drought Severity Index, PDSI).

Interventions

An *intervention* is a human action that could change the behavior of a system either in the past or in the future. Interventions are typically carried out by humans changing an existing situation (e.g., reducing the amount of agriculture subsidies) with the goal of changing outcomes. Interventions can be done at a given time point, or be sustained throughout some time period. We do not formally represent the intervention itself as an action, but rather we focus on representing the variables that it may affect.

Interventions have a role in all types of questions: diagnostic (e.g., what was the effect of the help sent to South Sudan during the 2017 famine?), prognostic (e.g., what would happen if help was sent to South Sudan in 2018?) or counterfactual (e.g., what would have happened if help had been sent in the form of cash rather than food supply?).

An *intervention target* (or manipulated variable) is a variable that represents measurable direct results of the actions taken in the intervention. A single intervention may have several intervention targets.

An *intervention result indicator* is a modeling variable that the user identifies as useful to quantify the effects of an intervention.

An *intervention risk indicator* is a modeling variable that the user identifies as useful to quantify the indirect or unintended (side) effects of an intervention.

An *intervention formulation* is a choice of intervention targets, result indicators, and risk indicators. An intervention formulation can be a theme for a workflow ensemble.

An *intervention specification* consists of the input data and parameters needed to specify the values of the intervention targets both in their initial state and throughout the modeling time period (If applicable). Note that an intervention specification is a subset of all the input data and parameters specified in a realization specification.

An *intervention realization* is an intervention specification, a possible realization of it, and the values of intervention result and risk indicators.

An *intervention impact pathway* is a graph that links intervention targets to result and risk indicators through causal relations. Causality is a fundamental aspect of understanding interventions.

Interventions are often combined together, which would result in potentially interacting impact pathways to amplify their effects. An *intervention impact network* is a graph with multiple

interventions and their associated impact pathways which can potentially share the same variables.

Interventions often have an associated cost, as do the changes to risk indicators that they incur. A cost-benefit analysis would need to be done by adding cost variables and cost models.

In some cases, measuring the impact of an intervention may require the use of a reference realization without any intervention even if it has actually occurred. For instance, to study the impact of a cash intervention in 2017 in South Sudan, a reference realization would be needed where no cash was sent. The analyst can then decide whether the intervention had an impact over the "do nothing" alternative.

Interventions can either be tactical (the user reacts to a developing situation) or strategic (these decisions are long-term and require planning).

A Note on Scenarios, Uncertainty, and Sensitivity

There are a few terms that are hard to define formally but are important aspects of modeling.

Scenarios

We note that the term scenario is a high-level term that is an organizing concept that end users find useful. Given a question, the user would formulate one or more scenarios to analyze that question. “Scenario” does not have a formal definition. It is often used to refer to workflow ensembles.

Uncertainty

There are many sources of uncertainty in modeling. It would be useful to make appropriate distinctions for those sources, and have computational definitions that can be used to quantify uncertainty.

Uncertainty can be associated with all input values of any model, and must be propagated through the model and reflected in the model outputs. Specifying the uncertainty associated with each input variable is important so that iterative executions of a specific realization can be run with all possible input values within the extent of their uncertainty.

Sensitivity

Sensitivity analysis evaluates the impact of the uncertainty contained in each input value on the outputs. A variable that has large influence in the model may have high sensitivity to its uncertainty, even if that uncertainty itself is small, while a variable with high uncertainty but little influence on the outputs will have a low sensitivity. Evaluating sensitivity is an important step in

understanding the relationships between variables in a model ensemble and requires iterative executions of a realization to measure. Sensitivity analysis allows modelers to understand which variables' are important to specify to a high degree and which others can be varied over a wide range of possible values within their uncertainty without affecting the results as they move on to other realizations.