# An Interactive Environment for the Modeling and Discovery of Scientific Knowledge

Will Bridewell [a,1,*], Javier Nicolás Sánchez [a], Pat Langley [a,1], Dorrit Billman [a,1]

[a] *Computational Learning Laboratory, Center for the Study of Language and Information, Stanford University, Stanford, CA 94305 USA*

**Abstract**

Existing tools for scientific modeling offer little support for improving models in response to data, whereas computational methods for scientific knowledge discovery provide few opportunities for user input. In this paper, we present a language for stating process models and background knowledge in terms familiar to scientists, along with an interactive environment for knowledge discovery that lets the user construct, edit, and visualize scientific models, use them to make predictions, and revise them to better fit available data. We report initial studies in three domains that illustrate the operation of this environment and the results of a user study carried out with domain scientists. Finally, we discuss related research on modeling formalisms and model revision, and we suggest priorities for additional research.

*Key words:* Scientific modeling, Interactive knowledge discovery, Model revision

## 1 Background and Motivation

Models play a central role in science, in that they utilize general laws or theories to predict or explain behavior in specific situations. Models occur in many guises, but the more complex the phenomena for which they account, the more important that they be cast in some formal notation with an unambiguous

* Corresponding author. Tel: +1 (650) 494-3884. Fax: +1 (650) 494-1588.
*Email addresses:* willb@csli.stanford.edu (Will Bridewell), jsanchez@cs.stanford.edu (Javier Nicolás Sánchez), langley@isle.org (Pat Langley), dbillman@psych.stanford.edu (Dorrit Billman).
[1] Also affiliated with the Institute for the Study of Learning and Expertise, 2164 Staunton Court, Palo Alto, CA 94306 USA.

interpretation. Moreover, the advent of fields like Earth science and systems biology, which attempt to explain the behavior of complex systems in terms of interacting components, have increased the need for computational tools to aid model construction and use.

A variety of computational modeling tools already exist, although they are typically associated with particular fields. For example, STELLA (Richmond et al., 1987) provides a language and environment for creating quantitative models in terms of instantaneous and difference equations. This framework has been adopted widely in the Earth science community for use in ecosystem models. Similarly, MATLAB (The Mathworks, Inc., 1997) offers an alternative formalism and environment for specifying quantitative models that include instantaneous and differential equations. However, it is most popular in engineering circles that model complex artifacts like electric circuits.

Both these and other environments offer interactive tools that let users visualize the structure of models, run them as simulations, and examine their predictions. However, they provide at most limited facilities for using available data to generate or improve models. That is, current modeling environments are concerned primarily with the formulation and simulation of models, not with their discovery. However, as data become more available and as the complexity of models grows, scientists would increasingly stand to benefit from such computational assistance.

On another front, there has been considerable research on computational methods for discovering knowledge from data. Much of this work, especially within the *data mining* paradigm (e.g., Fayyad et al., 1996), has emphasized formalisms like decision trees and logical rules that came originally from the field of artificial intelligence. These notations are perfectly appropriate for business applications, since the corporate world lacks established ways to represent domain knowledge, but they are more poorly suited for scientific disciplines, which have a long history of formalisms for encoding knowledge.

Fortunately, an alternative paradigm, known as *computational scientific discovery* (e.g., Langley, 2000), has dealt instead with discovery of knowledge cast as numeric equations and other notations widely used in fields of science and engineering. Yet research in this framework shares with data mining an emphasis on automating the discovery process, so that, with few exceptions, the developed methods provide little support for interaction with human users. Another drawback is that these methods typically focus on discovering knowledge from scratch, and thus takes no advantage of scientists' existing knowledge about a domain.

Clearly, scientists would benefit from computational tools that combine the advantages of available modeling environments with the strengths of existing

Table 1
A quantitative process model of a simple ecosystem with one predator (*D. nasutum*) and one prey (*P. aurelia*).

```
model Predator_Prey;
variables aurelia{prey}, nasutum{predator};
observable aurelia, nasutum;
  process nasutum_decay;
    equations d[nasutum,t,1] = −1 * 1.2 * nasutum;
  process aurelia_exponential_growth;
    equations d[aurelia,t,1] = 2.5 * aurelia;
  process predation_volterra;
    equations d[aurelia,t,1] = −1 * 0.1 * aurelia * nasutum;
              d[nasutum,t,1] = 0.3 * 0.1 * nasutum * aurelia;
```

discovery methods. We envision a computational framework that lets a scientist formulate a model, generate predictions from that model, detect anomalies that indicate need for revisions, and semi-automatically alter the model in response. The scientist would devise the initial model and guide high-level decisions about refinement, with the computer handling predictions, fine-grained search, and other easily automated steps. This view fits well with Shneiderman's (2000) proposal for computational tools that support creative inquiry.

In this paper we introduce PROMETHEUS, an environment that supports interactive knowledge discovery in this manner. As we describe shortly, the system includes a formalism for specifying models and background knowledge in terms of quantitative processes, which play a role in many scientific accounts. The environment includes tools for constructing, visualizing, and editing such process models, for utilizing them in predictive simulation, and for constrained revision of models in response to observations, thus supporting their iterative refinement. We demonstrate these capabilities in the context of revising models from three domains related to Earth science and microbiology. We also report the results of a user study that we carried out with ecologists trained in model construction. In closing, we discuss related work on simulation and discovery, along with directions for future research in this area.

## 2 A Language for Process Models

Before a scientist can develop and evaluate scientific models, he must first be able to represent them. To this end, the PROMETHEUS environment provides a programming language for specifying formal models. As in other formalisms for expressing quantitative knowledge, variables and the equations that relate them play a central role. However, traditional mathematical models leave im-

plicit an important aspect of scientific knowledge—*processes*—whereas PROMETHEUS makes it an explicit part of its models. In fact, the notion of a process is the central organizing principle in the programming language, so we refer to programs written in this formalism as *process models*.

We borrow this distinctive characteristic from work in qualitative process theory (Forbus, 1984), in which a user describes a dynamic system by the general relationships among its parts. That representation was designed to support commonsense reasoning and knowledge, whereas we developed PROMETHEUS' language with deterministic scientific modeling in mind. Some work in qualitative process theory includes support for quantitative knowledge (Forbus and Falkenhainer, 1990), which brings this work closer to our own, but the emphasis still rests on the qualitative processes, with the quantitative information stored in a separate library. Since we built PROMETHEUS as a tool for developing mathematical models, the associated language places the quantitative information on equal grounds with the qualitative content.

To illustrate the use of process models, we appeal to a classic example of predator–prey interaction. Consider an ecosystem consisting of two protist species *Paramecium aurelia* and *Didinium nasutum*, wherein the latter preys upon the former. Jost and Ellner (2000) give a thorough analysis of this simple ecosystem using traditional modeling methods. We developed a model of this ecosystem on the same general model structure that guided their exploration, and we use this model to illustrate the process modeling formalism and to demonstrate the PROMETHEUS environment.

Table 1 shows the candidate process model for the protist ecosystem. The specification begins with the model name (here, Predator_Prey) and the variables it references. This model has two variables, aurelia and nasutum, that represent the population density of each species in the ecosystem. A type, used primarily during model revision, follows the variable's name. In this example, aurelia has the type *prey* and nasutum has the type *predator*. The model also declares both aurelia and nasutum to be observable, which means that they were measured at some point during system activity.

Following the variable definitions come descriptions of the model's processes. Here we have three processes that explain how the values of variables change over time. A process consists of a name (e.g., predation_volterra) and one or more equations, plus an optional set of conditions. The processes in Table 1 involve differential equations, although we will see examples of algebraic equations later in the paper. The processes in this model always apply, so they have no conditions to restrict their activity.

The first process, nasutum_decay, indicates the death of *D. nasutum*, in which the left-hand side of the equation specifies a first-order differential equa-

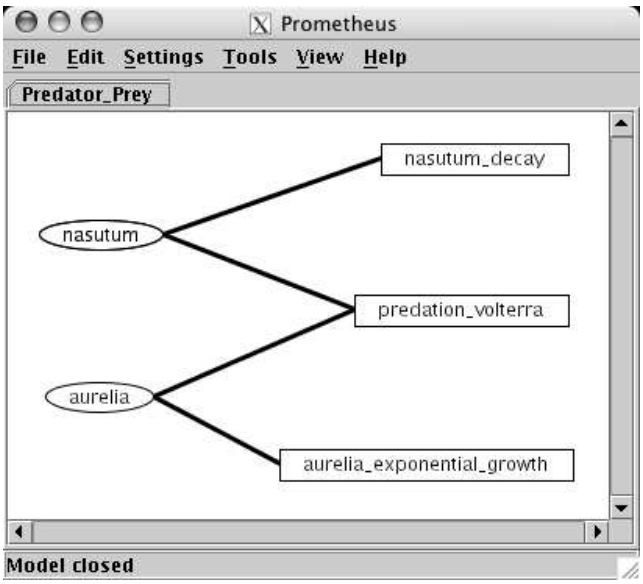


Fig. 1. PROMETHEUS' graphical display of the process model from Table 1.

tion for aurelia with respect to t (time) and the right-hand side indicates that density decreases $(-1)$ with a rate of 1.2. The second process, aurelia_exponential_growth, defines the growth of *P. aurelia*. The final process describes changes within both population densities, with the first equation defining the rate at which *D. nasutum* consumes *P. aurelia* and the second equation specifying the resulting increase in nasutum. When multiple processes influence the same variable, PROMETHEUS assumes the effects are additive, although other combining functions are possible.

## 3 Visualization and Simulation of Process Models

Once provided with a process model, PROMETHEUS lets the scientist visualize its causal structure. To illustrate, Figure 1 shows the graphical representation of the model from Table 1. The environment displays processes as rectangles and variables as ellipses. A thick line between a variable and a process, such as the one from nasutum to nasutum_decay, indicates that the variable appears on the left-hand side of a differential equation within that process. A thin line, not seen in this example, signifies that the variable participates as either input or output of the process. Additionally, when a clear causal ordering exists among variables, the environment places those variables serving as input to the causal process to the left of the variables affected by that process. This representation shows how the variables in the model interact. To examine the details of these interactions (e.g., the governing conditions and equations), the scientist can click on the corresponding process rectangle in the display.

In addition to displaying a model's causal structure, PROMETHEUS can simulate the model's behavior. To this end, the scientist must provide the values

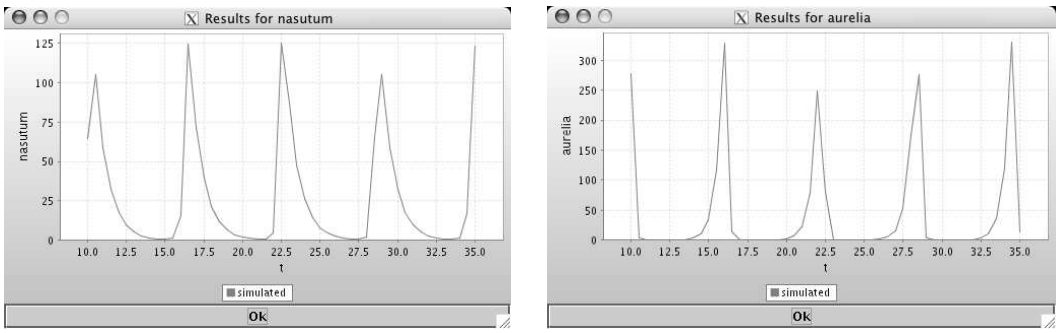


Fig. 2. Simulated trajectories for the predator–prey model from Table 1.

for each exogenous variable (i.e., those input variables that the model should not explain), initial values for each variable that occurs in the left-hand side of a differential equation, the start and end times of the simulation, and the size of the time step. In one such run, we used initial values from Jost and Ellner's (2000) analysis, [2] setting aurelia and nasutum to 276.60 and 64.67 individuals/mL, respectively. We also told the program to start the simulation on day 10, to end on day 35, and to report values once every 12 hours so that the timing of the predictions and the observed data would correspond.

After running the simulation, the user can view how a variable's predicted values change over time, as shown in Figure 2. PROMETHEUS draws a graph for the variable, with the x axis representing time and the y axis representing the variable's value. As the results indicate, the model produces a series of sharp peaks such that the growth of *D. nasutum* occurs slightly after the growth of *P. aurelia*. Once the prey population reaches its peak, a sharp decline precedes a similar decline in the predator population. To further evaluate a model, the user can plot the simulated results against the observed data. For example, Figure 3 shows how the process model's behavior compares to the data from Jost and Ellner's analysis. For both species the model produces fewer and sharper peaks than were observed, with the peaks being slightly out of synchronization with the observations.

Since the model fails to adequately reproduce the behavior of the ecosystem, the scientist may want to revise it. To this end, the graphical environment supports the addition and alteration of both variables and processes, [3] and the user has full access to the underlying text of the model. Thus the scientist can adjust the model, view the new causal structure, and simulate the new model's behavior. Used in this manner, PROMETHEUS acts primarily as a tool for model visualization and simulation, but it can also serve as an active

[2] These data are available at http://www.pubs.royalsoc.ac.uk/ as an appendix to Jost and Ellner's article. For this example, we used the data from their Figure 1(a) starting at day 10.

[3] The user can either instantiate a generic process from the library or create a new process from scratch.

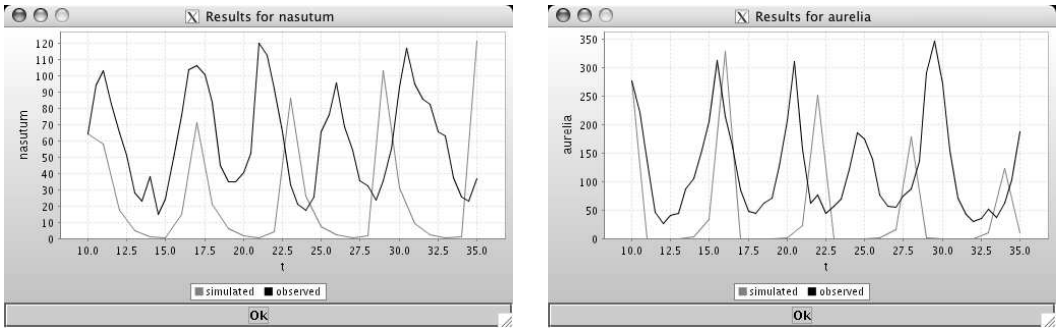


Fig. 3. Simulated versus observed output for the protist ecosystem.

assistant in the analysis of data as we will illustrate in the next section.

## 4 Revision of Process Models

When a model's predictions disagree with the observations, the PROMETHEUS user can also invoke a module for automated model revision. We can characterize the behavior of this system component as:

- *Given*: trajectories for a set of continuous variables over time;
- *Given*: background knowledge cast in terms of generic processes and variable type hierarchies;
- *Given*: observable and theoretical variables to be modeled;
- *Given*: an initial model, processes that may be removed from the model or have their parameters changed, generic processes that may be added to the model, and constraints on model size;
- *Find*: a set of process models, ranked by their fit to the data, that explain the observed trajectories.

In this section, we discuss the input to this approach in more detail, describe the search algorithm that produces the revised models, and show the output from the module.

Before PROMETHEUS can aid in model revision, it must have some knowledge about the domain. One type of knowledge takes the form of generic processes, which serve as building blocks when adding new processes to the model. Generic processes specify the form of a model's instantiated processes and have an analogous representation. Table 2 shows five generic processes relevant to modeling predator–prey interaction. Each generic process consists of five components: a name, a set of variables, a set of parameters, a set of

Table 2
Some generic processes relevant to the predator–prey model in Table 1.

```
generic process logistic_growth;
  variables S{prey};
  parameters p[0,3], k[0,1];
  equations d[S,t,1] = p * S * (1 − k * S);
generic process predation_volterra;
  variables S1{prey}, S2{predator};
  parameters a[0,1], b[0,1];
  equations d[S1,t,1] = −1 * a * S1 * S2;
            d[S2,t,1] = b * a * S1 * S2;
generic process predation_holling;
  variables S1{prey}, S2{predator};
  parameters a[0,1], b[0,1], c[0,1];
  equations d[S1,t,1] = −1 * a * S1 * S2/(1 + c * a * S1);
            d[S2,t,1] = b * a * S1 * S2 / (1 + c * a * S1);
generic process exponential_growth;
  variables S{prey};
  parameters b[0,3];
  equations d[S,t,1] = b * S;
generic process exponential_decay;
  variables S{species};
  parameters a[0,2];
  equations d[S,t,1] = −1 * a * S;
```

conditions, and a set of equation forms. Of these components, the parameters and conditions are optional. To instantiate a generic process, the user must provide both variables of the correct type and parameters that fall within a specified range. For example, aurelia_exponential_growth in Table 1 instantiates exponential_growth so that aurelia fills the role of $S$ and $b$ equals 2.5.

In addition to generic processes, the scientist must provide a type hierarchy over the variables. The types *predator* and *prey* from the protist ecosystem model are subtypes of *species*, which in turn is a subtype of *number*—the root of the hierarchy. When instantiating a generic process, PROMETHEUS must select variables of the appropriate types. Consider exponential_decay, which expects a species variable. Since both aurelia and nasutum are instances of species, either one can fill the role. In contrast, predation_volterra requires one variable each of the more specific types predator and prey. Here, knowledge of the variable types keeps PROMETHEUS from considering implausible models in which predator and prey switch roles.

Along with this general domain knowledge, the scientist provides PROMETHEUS with additional, task-specific information to direct its search for alternative models. This information includes a set of variables to include in the

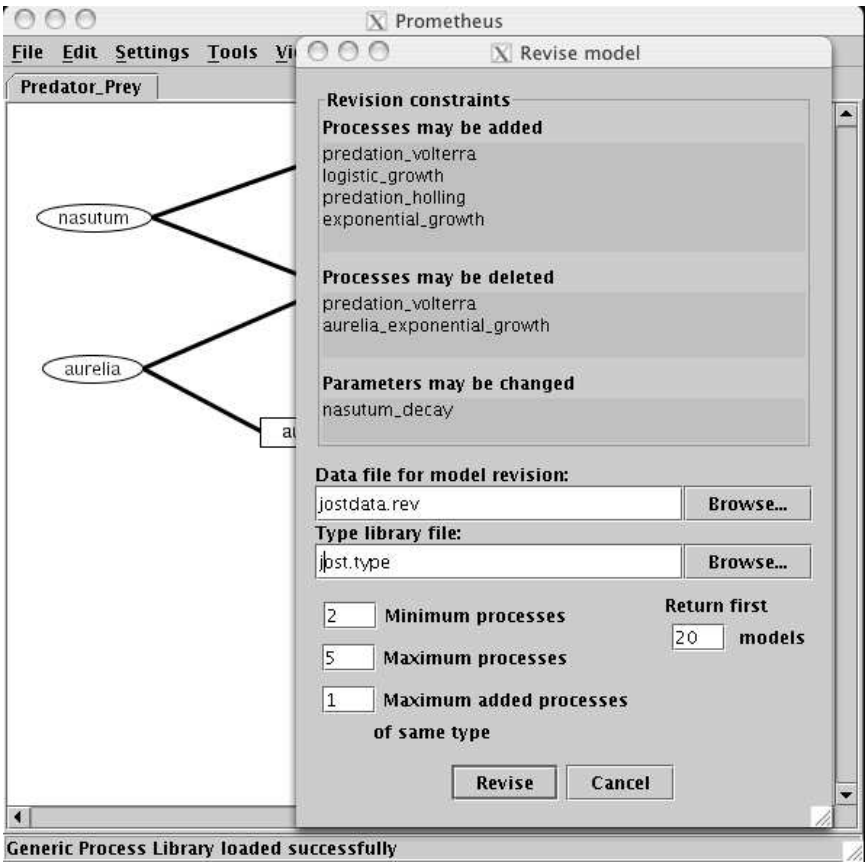


Fig. 4. Parameters used to revise the model of the protist ecosystem.

model, a data set containing values for the observable variables, and guidelines concerning the modification of the model. Specifically, the scientist can select which generic processes should be considered for addition and which current processes can be deleted or tuned by altering their parameters. Additionally, he can place limits on both the total number of processes in the model and the number of instantiations allowed for each generic process.

Figure 4 shows settings given to PROMETHEUS to guide its revision of the model in Table 1. The top portion of the dialog box indicates that the program should consider adding instantiations of predation_volterra, predation_holling, logistic_growth, and exponential_growth. The user also told PROMETHEUS to consider deleting the current aurelia_exponential_growth and predation_volterra processes, and to alter the parameters of nasutum_decay. Additional input specifies that the resulting model should contain from two to five processes and only one instantiation of each generic process.

Once provided with this information, PROMETHEUS searches for a revised model using the method described by Asgharbeygi et al. (2006). First, the program removes the deletable processes from the model and sets them aside, which produces a model containing both the immutable processes and the

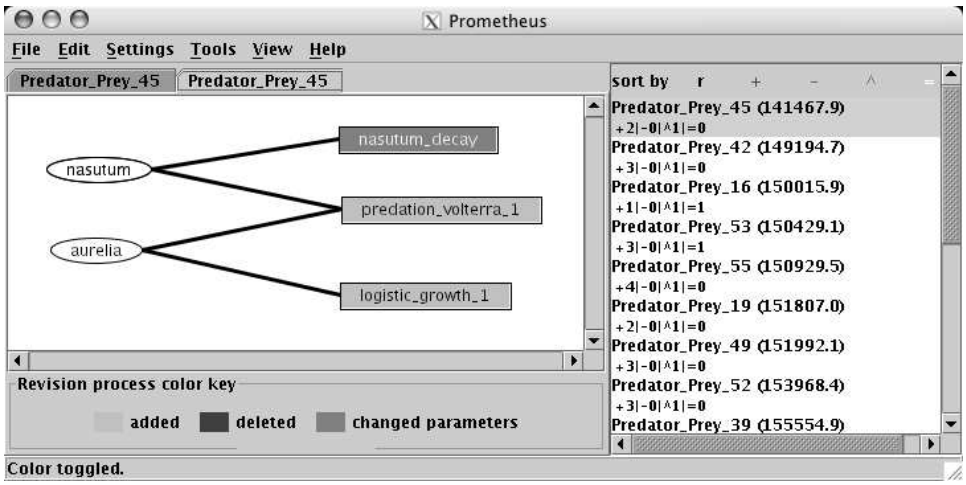


Fig. 5. Revised models for the protist ecosystem, ranked by the sum of squared error, along with the structure of the best fitting candidate, as displayed in PROMETHEUS.

processes with alterable parameters. Second, PROMETHEUS binds variables by type to the generic processes that may be added to the model. This step can lead to multiple instantiations of the same generic process. For example, if there were two different prey in the protist ecosystem, then two logistic_growth processes from Table 2 could be created. Third, the program places both the deletable and the freshly bound processes into a pool of addable components.

After dividing the input into required and optional processes, PROMETHEUS carries out a two-stage search through the model space. In the first stage, the program creates candidate structures by first generating all subsets of the component pool and then adding each one to the base model. To remain viable, each model structure must satisfy five constraints:

- the number of processes must fall within the user-specified bounds;
- a cycle composed solely of algebraic equations cannot exist;
- exogenous variables cannot act as the output of any process;
- any observable variable acting as an input to a process must also act as the output of another process; and
- each unobservable variable must be both the output of some process and the input to another process.

For each structure, PROMETHEUS performs a gradient-descent search through the parameter space defined by the new processes, those marked as changeable, and the initial values of the variables. The search algorithm is an iterative procedure with an external loop that first, selects a random starting point and uses the Levenberg–Marquardt method (Levenberg, 1944; Marquardt, 1963) to find a local optimum. An internal loop makes random jumps from this optimum along the dimensions of the parameter vector. If a jump results in a lower error score, PROMETHEUS calls the Levenberg-Marquardt method from

Table 3
The most accurate revised model for the protist ecosystem.

```
model Predator_Prey_revised;
variables nasutum{predator},aurelia{prey};
observable nasutum,aurelia;
  process logistic_growth_1;
    equations d[aurelia,t,1] = 1.810082 * aurelia * (1 − 0.000288 * aurelia);
  process predation_volterra_1;
    equations d[aurelia,t,1] = −1 * 0.03002 * aurelia * nasutum;
              d[nasutum,t,1] = 0.292278 * 0.03002 * aurelia * nasutum;
  process nasutum_decay;
    equations d[nasutum,t,1] = −1 * 1.034667 * nasutum;
```

this point and restarts the internal loop. When this strategy fails to reduce the error, the environment restarts the iterative procedure from another randomly selected point. The search ends after a fixed number of random restarts.

Once PROMETHEUS completes the search, it returns the set of $n$ models with the smallest error score. In this example, the program returns for the scientist's inspection the 20 model structures, with their best parameter estimates, that result in the lowest sum of squared error. Figure 5 shows how the environment displays the revisions. The list of models appears on the right, each with its name, sum of squared error, and the number of processes that were added (+), deleted (-), changed (^), and unchanged (=). The user can inspect a model's details by selecting it on the screen. In addition, each model's components are color coded to ease the identification of altered processes. Figure 5 displays the causal structure of the model with the smallest error on the protist data.

Table 3 shows this best scoring model, which differs from the initial one in three ways. At the structural level, aurelia_exponential_growth has been replaced by a logistic process so that population growth now slows once a certain density has been reached. Within processes, the rate of change is lower in both equations associated with predation_volterra, and the process nasutum_decay now has a slower death rate for *D. nasutum*.

Figure 6 compares the trajectories for *D. nasutum* and *P. aurelia* density produced by the new model with the original data. In both species, the number of peaks and the synchronization of the oscillations match the observations much more closely than before. Although peak heights have also improved, the model does not account entirely for the peak occurring in both populations on the thirtieth day. However, using PROMETHEUS, the scientist can refine this model further to improve its behavior compared to that observed in the protist ecosystem.

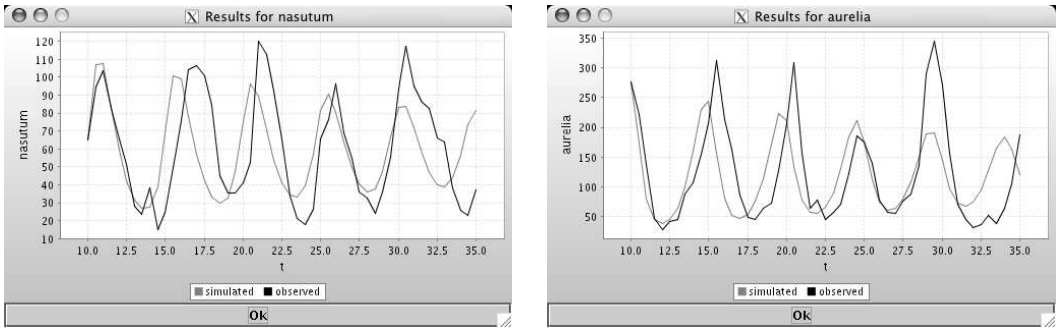


Fig. 6. Simulated trajectories predicted by the revised protist model and observed values for the same system.

## 5 Modeling an Aquatic Ecosystem Using PROMETHEUS

In evaluating PROMETHEUS, we have also modeled the Ross Sea ecosystem, which Arrigo et al. (2003) have described at length. For this system, scientists are particularly interested in the change in phytoplankton population throughout the year. Suspected influences include the availability of nutrients and light, as well as grazing behavior by zooplankton. Table 4 shows an initial process model for this ecosystem.

As in the protist model, variables appear first. In this case, zooplankton (zoo) and phytoplankton (phyto) indicate two species, nitrate is the primary nutrient for the phytoplankton, and both light and ice are pertinent environmental factors. Of these variables, only phyto, nitrate, and ice are observable; these denote the measured concentrations of phytoplankton, $NO_3$, and sea ice, respectively. In addition to being observable, ice is exogenous, meaning both that the model should not explain its behavior and that the scientist must provide the values of this variable when simulating or revising the model.

The process definitions follow the list of variables. While most processes in this model are relatively straightforward, set_constants is distinctive in that it shows how, within our formalism, the user can define parameters that are shared among multiple processes. Since our language revolves around variables and processes, the user treats the constants as variables, placing equations that specify the values inside a process. Thus we need not introduce new language structures to represent global parameters. As with the parameters local to a process, the values of these constants can be tuned during model revision.

The process for light_production clarifies another important feature of the environment—algebraic equations can express instantaneous effects. In practice, PROMETHEUS computes the values of these equations immediately after it simulates the differential equations for a particular point. The algebraic equation within light_production indicates that sunlight varies as a function of the day of the year. Since the Ross Sea is deep in the Southern Hemisphere,

Table 4
A quantitative process model of the Ross Sea ecosystem.

```
model Ross_Sea_Ecosystem;
variables zoo{z_species}, nitrate_to_carbon_ratio{n_const},
          light{signal}, nitrate{n_nutrient}, phyto{p_species},
          ice{fraction}, light_rate{l_rate}, G{gz_rate},
          growth_rate{gw_rate}, nitrate_rate{n_rate},
          remin_rate{r_rate}, r_max{r_const}, residue{residue};
observable nitrate, phyto, ice;
exogenous ice;
  process light_production;
    equations light = max(0.5 * 410 * cos(6.283 * t / 365), 0) * ice;
  process phyto_loss;
    equations d[phyto,t,1] = −0.1 * phyto;
              d[residue,t,1] = 0.1 * phyto;
  process phyto_growth;
    equations d[phyto,t,1] = growth_rate * phyto;
  process phyto_absorption_nitrate;
    equations d[nitrate,t,1] = −1 * nitrate_to_carbon_ratio * growth_rate * phyto;
  process growth_limitation;
    equations growth_rate = r_max * min(nitrate_rate,light_rate);
  process nitrate_availability;
    equations nitrate_rate = nitrate / (nitrate + 5);
  process light_availability;
    equations light_rate = light / (light + 50);
  process set_constants;
    equations nitrate_to_carbon_ratio = 0.251247;
              r_max = 0.193804;
              remin_rate = 0.067559;
```

the periods of day and night are extended. The equation produces cycles in rough accordance with the natural availability of sunlight while ensuring that values never become negative. The process multiplies the light intensity by the ice concentration because the ice particles reduce the availability of light to the phytoplankton.

Figure 7 shows how PROMETHEUS displays this model graphically. The top portion indicates that the concentration of ice affects the available light and hence the growth rate of phytoplankton. Similarly, the next chain of influence down relates $NO_3$ to phytoplankton's growth. The concentration of phytoplankton itself is a direct result of the process governing its growth and the process governing its loss. Two variables, zoo and G, which refer to zooplankton's concentration and growth rate, are unconnected, indicating that this model lacks grazing effects.

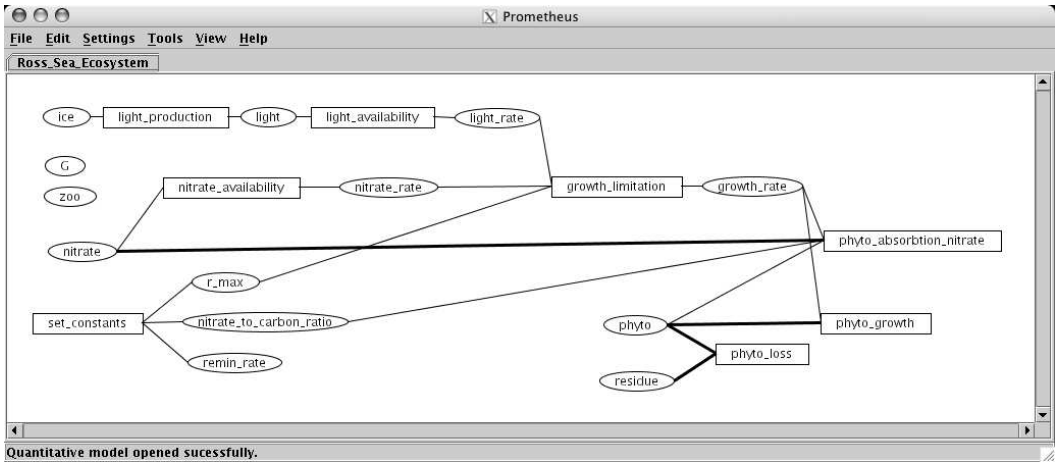


Fig. 7. The graphical representation of tje Ross Sea ecosystem model from Table 4 as displayed in PROMETHEUS.

Figure 8 compares the change over time in phytoplankton and nitrate concentrations as simulated by this model to that actually observed. Although a slight increase appears in the phytoplankton trajectory, it comes too late in the season, after light availability has already diminished. Therefore, the model does not reproduce the exponential growth followed by an exponential decrease that actually occurred in the Ross Sea. However, when the phytoplankton population increases, less nitrate is available.

To revise the model so that it better fits the data, we invoked PROMETHEUS' revision component. As in the predator–prey example, we provided types for the variables and a list of generic processes. Table 4 shows the types in the original model, whereas the generic processes that we let PROMETHEUS instantiate and add to this model appear in Table 5. In addition, we let the environment alter the parameters of all current processes except for light_availability, light_production, and set_constants.

Table 6 presents the alterations to the original model that occurred in the best-scoring revision. PROMETHEUS added one each of the generic processes in

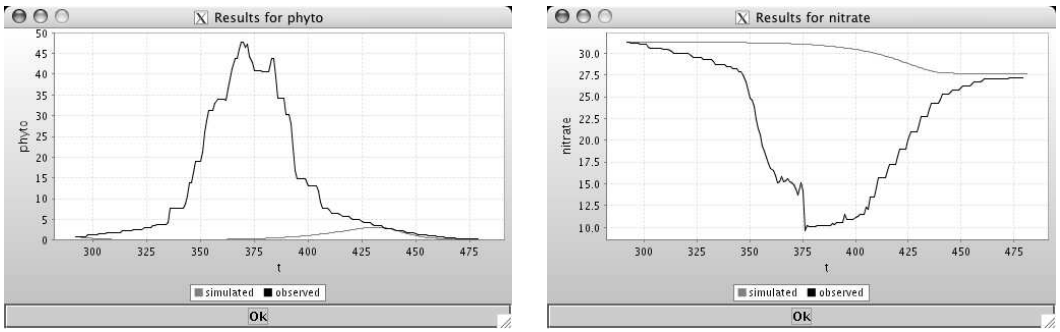


Fig. 8. Observed concentrations of phytoplankton and nitrate, along with values predicted by the initial model of the Ross Sea ecosystem.

Table 5
Generic processes used for revising the model of the Ross Sea ecosystem.

```
generic process zoo_grazes_phyto;
   variables P{p_species}, Z{z_species}, R{residue}, G{gz_rate};
   parameters gamma[0,1];
   equations d[P,t,1] = −1.0 * G * Z;
             d[R,t,1] = gamma * G * Z;
             d[Z,t,1] = (1 − gamma) * G * Z;
generic process Ivlev_rate;
   variables G{gz_rate}, P{p_species};
   parameters delta[0,10],rho[0,10];
   equations G = rho * (1 − exp(−1 * delta * P));
generic process residue_loss_to_remineralization;
   variables RES{residue}, REM{r_rate};
   equations d[RES,t,1] = −1 * REM * RES;
generic process nitrate_remineralization;
   variables N{n_nutrient}, REM{r_rate}, RES{residue}, NtoC{n_const};
   equations d[N,t,1] = REM * NtoC * RES;
```

Table 5 and altered the parameters of phyto_loss and nitrate_availability. The new processes zoo_grazes_phyto_1 and Ivlev_rate_1 jointly characterize zooplankton's grazing on phytoplankton, while residue_loss_to_remineralization_1 and nitrate_remineralization_1 describe the restoration of nitrate to the environment that results from the death and decay of phytoplankton. Additionally, the parameter in nitrate_availability increased and the one controlling phyto_loss decreased.

Figure 9 presents the results of simulating the revised model and their relation to the observed data.[4] As can be seen, this model conforms to the observed data much better than the original version. The modeled growth of phytoplankton now peaks at the right time and magnitude, while the nitrate concentration also changes in roughly the correct fashion. However, discrepancies still exist between the predicted and observed trajectories. Most notably, the initial increase in phytoplankton concentration grows more slowly than observed, and the nitrate concentration decreases more than it should. However, the revisions produced by PROMETHEUS let one concentrate on these secondary features of the ecosystem, giving a more appropriate starting point for fine-grained analysis.

[4] In addition to fitting parameters in the differential equations, PROMETHEUS also selects initial values for the variables in the revised model. These values account for the discrepancy in starting points for the simulated versus observed trajectories.

Table 6
Processes that were either altered or added by PROMETHEUS to the original Ross Sea model.

```
process zoo_grazes_phyto_1;
  equations d[phyto,t,1] = −1 * G * zoo;
            d[residue,t,1] = 0.914228 * G * zoo;
            d[zoo,t,1] = (1 − 0.914228) * G * zoo;
process Ivlev_rate_1;
  equations G = 2.232819 * (1 − exp(−1 * 0.004399 * phyto));
process residue_loss_to_remineralization_1;
  equations d[residue,t,1] = −1 * remin_rate * residue;
process nitrate_remineralization_1;
  equations d[nitrate,t,1] = remin_rate * nitrate_to_carbon_ratio * residue;
process phyto_loss;
  equations d[phyto,t,1] = −0.017099 * phyto;
            d[residue,t,1] = 0.017099 * phyto;
process nitrate_availability;
  equations nitrate_rate = nitrate / (nitrate + 9.804389);
```

## 6 Modeling Photosynthesis Regulation with PROMETHEUS

In addition to the two ecosystems already described, we used PROMETHEUS to investigate the regulation of photosynthesis. Biologists have devoted a large amount of time to this system, and they continue to investigate the underlying mechanisms. Recently, Labiosa et al. (2003) examined photosynthetic behavior within the cyanobacterium *Synechocystis* sp. PCC 6803. Their experiments simulated natural lighting conditions and sampled the bacteria at nine points within a 24-hour period. They processed these samples using cDNA microarray technology, and measured mRNA concentrations for numerous genes. We used PROMETHEUS to build a plausible model of photosynthesis regulation, to analyze the microarray data, and to revise this model.

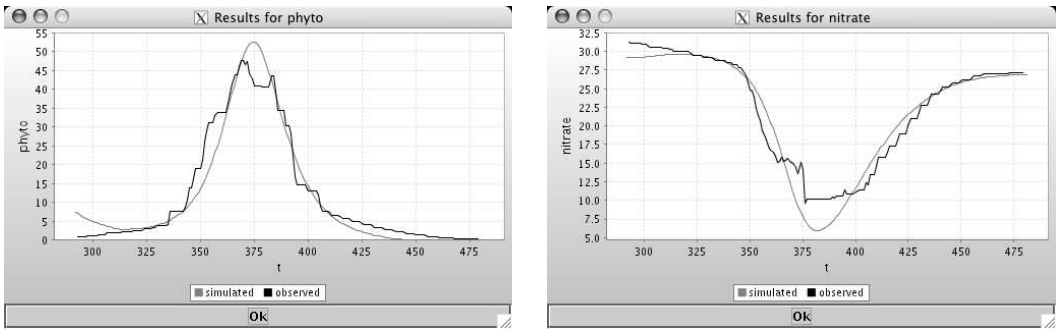


Fig. 9. Observed concentrations of phytoplankton and nitrate, along with values predicted by the revised model of the Ross Sea ecosystem.

Table 7 displays the initial model of photosynthesis regulation, which relates six variables. The first represents the amount of light available to the observed plants throughout the day. As in the Ross Sea model, the amount of light is simulated using a trigonometric function, which ensures that the light intensity peaks at noon. The next three variables represent concentrations of mRNA, photosynthetic protein, and reactive oxygen species (ROS), respectively. The mRNA variable encodes an aggregate over 17 genes that were implicated in regulation of the photosynthetic system, whereas both photosynthetic protein and ROS are biologically plausible theoretical terms. The former denotes the average concentration of all proteins involved in photosynthesis, whereas the latter represents the amount of a damaging byproduct the process produces. The final two variables signify the amount of energy in the system (redox) and the rate of mRNA transcription.

The eight processes in the initial model are similar in form to those we have previously discussed. Photosynthesis produces both redox and ROS. Translation increases the amount of protein, while transcription increases the mRNA concentration while consuming redox. The negative effect of ROS on protein is captured in protein_degradation_ros, and the normal degradation of mRNA is represented by mRNA_degradation.

Unlike the other models, some of the processes in Table 7 have conditions, which are stated as arithmetic relations placed before the equations and separated by commas. During simulation, a process is active only when all of its conditions are met. As an example, mRNA_degradation cannot occur unless mRNA is present, so the model explicitly requires a positive mRNA concentration for the degradation process to proceed.

Figure 10 shows how PROMETHEUS displays the process model for photosynthesis regulation. This graphical representation reveals three primary pathways that are tied together by three processes. The lighting pathway provides input to photosynthesis and affects the amount of mRNA by influencing the transcription rate. The pathway containing ROS and photosynthetic protein describes how protein concentrations decrease within the cell as affected by the amount of mRNA through photo_translation. The last pathway describes the change in mRNA due to the amount of cellular energy. All three interact through the central photosynthesis process that uses light and produces both ROS, which lowers the protein concentration, and redox, which increases mRNA transcription.

Figure 11 presents the simulated results from the model compared with the observed mRNA values. As in the data, the predicted trajectory has two peaks, with a striking drop in mRNA concentration at noon. However, both the magnitude and timing of the events are incorrect. The first peak produced by the model occurs too early in the day, and the last peak both occurs too late

Table 7
The initial model of photosynthesis regulation.

```
model Photosynthesis_Regulation;
variables light{light}, mRNA{mRNA}, transcription_rate{rate},
          ROS{ros}, redox{redox}, photo_protein{photo_protein};
observable mRNA;
  process photosynthesis;
    equations d[redox,t,1] = 1.50 * light * photo_protein;
              d[ROS,t,1] = 1.00 * light * photo_protein;
  process photo_translation;
    equations d[photo_protein,t,1] = 0.20 * mRNA;
  process protein_degradation_ros;
    conditions photo_protein > 0, ROS > 0;
    equations d[photo_protein,t,1] = −0.05 * ROS;
              d[ROS,t,1] = −0.05 * ROS;
  process mRNA_transcription;
    equations d[mRNA,t,1] = transcription_rate;
  process regulate_light;
    equations transcription_rate = 0.80 * light;
  process regulate_redox;
    conditions redox > 0;
    equations transcription_rate = −2.00 * redox;
              d[redox,t,1] = −1.00 * redox;
  process mRNA_degradation;
    conditions mRNA > 0;
    equations d[mRNA,t,1] = −0.02 * mRNA;
  process lighting;
    equations light = 1 − cos((2 * 3.1415926 / 24) * t);
```

and overshoots the observed maximum concentration. Even more distressing is that the concentration of mRNA dips below zero for several hours. Each of these discrepancies in the simulated trajectory indicates that we should attempt to revise the model.

In earlier work (Langley et al., 2006), we reported the model in Table 8, which provides an improved fit to the data. Structurally, this model differs from that in Table 7 due to the absence of regulate_light. Originally, the amount of light affected mRNA translation directly, but the revised version posits that light has only an indirect effect due to its influence on redox. In addition to this structural change, PROMETHEUS altered the parameters of all the processes. The resulting model leads to the behavior shown on the right in Figure 11, which fits the observations almost perfectly. [5]

[5] We have not reported the observed data because our biologist collaborators have not yet published them. Also, given the noise inherent in microarrays, such a good

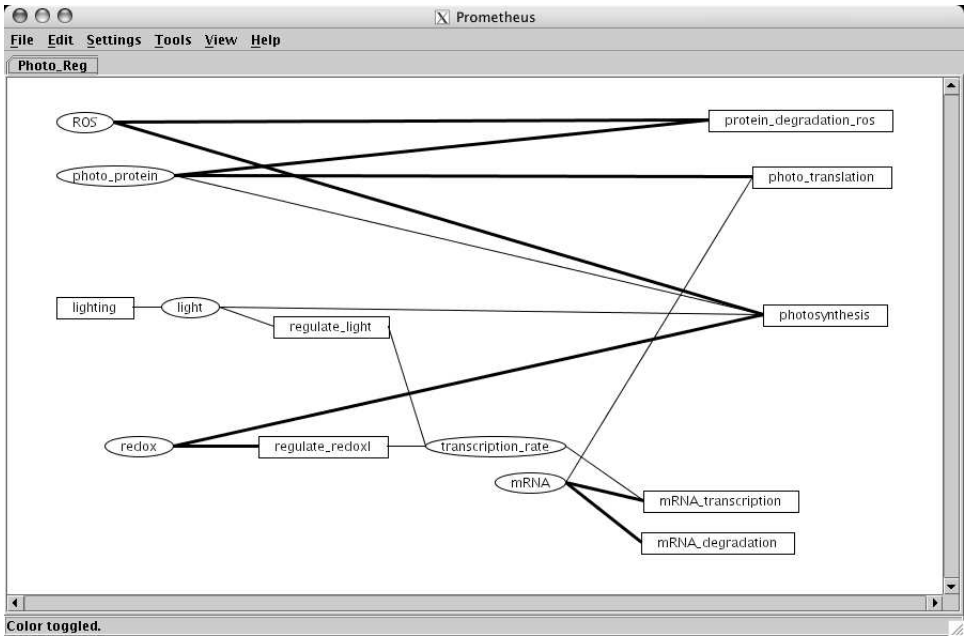


Fig. 10. PROMETHEUS' display of the photosynthesis model from Table 7.

## 7 Feedback from Users

While developing PROMETHEUS, we interacted with ecologists to get feedback about the utility of the program. Two of these scientists belong to our research team: one is a prominent oceanographer who both develops models and trains students in his department; the other is a postdoctoral ecologist who spends most of his time working directly with the PROMETHEUS developers. Their input has been invaluable to improving the system's usability and incorporating meaningful domain knowledge, but we wanted additional evidence about the environment's suitability as a computational aid to model development.

To this end, we carried out a pilot study to obtain input from two graduate students and a senior technician, all from the oceanographer's research group. Each user had experience with building ecological models, but this had involved writing programs in FORTRAN. To gather feedback, we introduced users to the system's capabilities, interface, and modeling conventions. We then asked them to revise a model of the Ross Sea to produce a better fit to simulated data for four variables. Changes to both the model's structure and parameters were necessary to successfully complete the task, and each participant discovered the generating model. This result indicates that even though they found the system unfamiliar, the users could learn to use it productively

match suggests that we are overfitting the training set, but our point was to demonstrate another domain for which our approach is relevant, not to propose this as the correct model.

Table 8
A revised model of photosynthesis regulation that PROMETHEUS produced.

```
model Photosynthesis_Regulation;
variables light{light}, mRNA{mRNA}, transcription_rate{rate},
          ROS{ros}, redox{redox}, photo_protein{photo_protein};
observable mRNA;
  process photosynthesis;
    equations d[redox,t,1] = 3.62 * light * photo_protein;
              d[ROS,t,1] = 1.34 * light * photo_protein;
  process photo_translation;
    equations d[photo_protein,t,1] = 0.05 * mRNA;
  process protein_degradation_ros;
    conditions photo_protein > 0, ROS > 0;
    equations d[photo_protein,t,1] = −1 * 0.10 * ROS;
              d[ROS,t,1] = −1 * 0.10 * ROS;
  process mRNA_transcription;
    equations d[mRNA,t,1] = transcription_rate;
  process regulate_redox;
    conditions redox > 0;
    equations transcription_rate = −12.72 * redox;
              d[redox,t,1] = −1 * 5.31 * redox;
  process mRNA_degradation;
    conditions mRNA > 0;
    equations d[mRNA,t,1] = −1 * 0.82 * mRNA;
  process lighting;
    equations light = 1 − cos((2 * 3.1415926 / 24) * t);
```

within a single, three-hour session.[6] To document user comments and actions, we recorded the sessions on video, captured screen activity with software, and elicited opinions using a survey. As we analyzed the feedback, we placed it into three categories: the visual environment, the supported tasks, and the underlying formalism.

PROMETHEUS' interface is relatively basic, and users had a variety of comments about how to improve it. Some of their suggestions emphasized modifications in the menu-driven steps needed to carry out revision, the conventions and display of model names, and the directness of accessing the generic process library. In addition, the edge semantics in the graphical display provided a common source of confusion. For instance, when a variable serves as both input and output to differential equations in a process, PROMETHEUS only shows the edge from the process to the variable. In response, we intend to

[6] The participants used a slightly different version of PROMETHEUS than we have detailed here, with a modified menu and window layout.

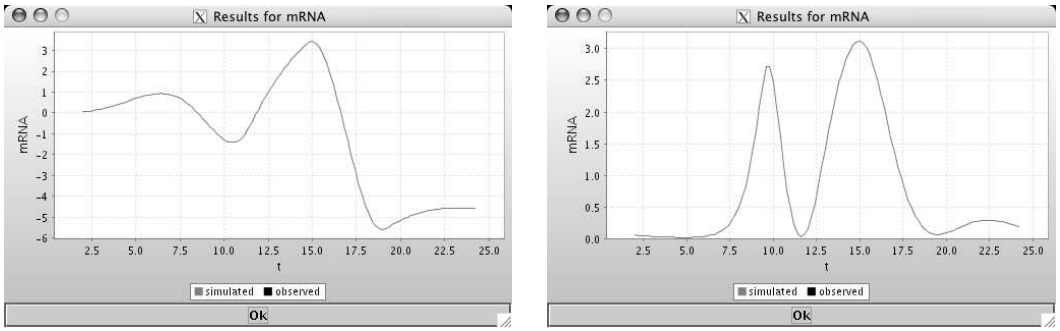


Fig. 11. Simulated and observed trajectories of mRNA concentration using the original (left) and revised (right) model of photosynthesis regulation. The observed trajectories are not shown.

modify the display to make the relations between processes and variables explicit without increasing visual clutter.

As an integrated modeling environment, PROMETHEUS supports multiple tasks, some of which were novel to the users. The participants in our study had little difficulty familiarizing themselves with the various operations supported by the program, and they expressed interest in the automated revision module. Users also appreciated the inclusion of the simulation and plotting tools, since model evaluation in ecology extends beyond a single measure of fitness. They also liked the ability to alter the model at an abstract, process-oriented level instead of editing equations or FORTRAN code. Indeed, they wanted the functionality deepened because many of their modeling tasks include partial differential equations; thus, for PROMETHEUS to be useful in their current research, we must extend it to support spatial models. However, the participants did mention that the existing environment would be useful for other ecological tasks and for teaching.

A key claim about PROMETHEUS and the process modeling formalism is that domain scientists can understand models built within the environment. While developing the program, our primary source of feedback was from the oceanographer in the group. After we explained the syntax of the language and the graphical layout, he had no difficulty understanding the model at either level. During informal discussions, he could also map generic processes to pieces of his working model of the Ross Sea (Arrigo et al., 2003). Although each participant in the user study spent some time looking at the underlying representations of the processes, only one wanted to examine the process library in detail. He expressed some surprise at the organization of the processes, but he had no difficulty interpreting the formalism for expressing processes or mapping between process and equation representations. The primary source of confusion in understanding the process modeling language appeared to be familiarity with the stock-and-flow approach to modeling supported by STELLA (Richmond et al., 1987). Subjects were accustomed to edges that indicate material

flow between variables as opposed to causal influences. As a result, we plan to incorporate a second graphical view that better matches stock-and-flow diagrams, thus providing a familiar representation to help ecologists interpret their models.

The user study provided generally encouraging results, but it also suggested extensions that should make PROMETHEUS more useful to domain scientists. In summary, the users appreciated the ability to search through a space of model structures and liked the integration of multiple model-related tools. However, their comments about spatial modeling and the visual representation of the model indicated obvious areas for improvement.

## 8 Related Research on Modeling and Discovery

In the preceding text, we illustrated both the structure of quantitative process models and the capabilities of PROMETHEUS. We introduced a formalism that lets a scientist represent mechanisms as networks of variables and familiar processes, along with an environment that displays the causal structure of the model and simulates its behavior. If the simulated trajectories fail to match observed data, the user can ask PROMETHEUS to propose revisions that improve its fit in ways consistent with domain knowledge. Generic processes provide the link in these revision efforts, giving the ability to produce explanatory models, as opposed to simply descriptive ones. This combination of features distinguishes the system from other quantitative modeling environments.

As we indicated in the introduction, PROMETHEUS' approach to scientific modeling is not entirely new, but rather borrows ideas from two previously disconnected literatures. However, it does more than simply combine two existing technologies; it moves beyond them to demonstrate new functionality and address new issues in interface design. Here we discuss in more detail the relations between our approach and earlier work.

On the one hand, the PROMETHEUS environment has many similarities to modeling frameworks like STELLA (Richmond et al., 1987) and MATLAB (The Mathworks, Inc., 1997). They share the notion of a formal syntax for specifying both instantaneous and dynamic quantitative models in terms of mathematical equations, although their detailed notations differ. In addition, they let the user create and edit models in this syntax, as well as invoke an associated simulator that can run those models to generate predictions. Finally, they all provide a graphical interface that lets the user display and inspect the logical structure of his mathematical models. Our approach also shares many features with Keller's (1995) SIGMA, which is another graphical environment that takes an interactive approach to model building, visualization, and anal-

ysis, as well as providing extensive checks to ensure model consistency and to handle unit conversions.

However, PROMETHEUS moves beyond these earlier modeling environments by requiring the user to organize equations into *processes*. This idea plays a central role in many scientific disciplines and appears in qualitative modeling tools such as HOMER (Bredeweg and Forbus, 2003), but previous quantitative simulation languages have not supported it. Equally important, the new environment supports computational revision of models in response to data, constrained by domain knowledge in the form of generic processes and by input from the user. MATLAB includes some facilities for attempting to estimate a model's parameters for a given data set, but it cannot alter the basic structure of a model.

At the same time, PROMETHEUS incorporates many ideas from earlier work on computational scientific discovery. In particular, it adopts the metaphor of heuristic search through a space of candidate structures guided by their ability to fit data. Our approach differs from other quantitative discovery work (e.g., Langley et al., 1987; Washio and Motoda, 1998) by focusing on process models, rather than on independent sets of equations, and by emphasizing revision of models rather than their generation, although it borrows ideas on this front from other efforts. Early research in this area emphasized qualitative models (e.g., Ourston and Mooney, 1990; Towell, 1991), although some more recent work addresses quantitative models with numeric equations (e.g., Chown and Dietterich, 2000; Saito et al., 2001; Todorovski and Džeroski, 2001).

The environment also differs from most earlier discovery research by its reliance on explicit domain knowledge to constrain search. Easley and Bradley (1999) take a similar approach in that their system uses *generalized physical networks*, which take the form of general equations, as background knowledge to discover differential equation models of nonlinear dynamic systems. Similarly, Todorovski and Džeroski's (1997) LAGRAMGE encodes background knowledge in terms of context-free grammars that specify the space of equations to consider during its search for models. PROMETHEUS draws on a similar idea, but employs domain knowledge in terms of generic processes rather than these other formalisms, as has Todorovski's (2003) recent work.

But the main difference from earlier discovery research concerns the interactive nature of our environment. Previous work on computational scientific discovery has focused almost exclusively on automated methods, whereas PROMETHEUS aims explicitly to support scientists rather than to replace them. This philosophy is consistent with a general trend in artificial intelligence research toward advisory systems, but it means we have had to address issues about human–computer interaction (e.g., how best to let users constrain the search for revised models) that some algorithm-oriented researchers will find unin-

teresting. Nevertheless, such issues must receive serious attention if we hope to develop computational discovery tools that practicing scientists will use on a regular basis.

We should note that our environment is not quite the first designed to accept user input.[7] For example, Valdés-Pérez (1995) developed MECHEM, which finds chemical reaction pathways that explain how a set of reactants produce a set of observed products. In addition to background knowledge about catalytic chemistry, the system accepts input from the user about constraints, expressed in terms familiar to chemists, that the inferred pathways must satisfy. The user can only influence MECHEM's behavior by setting switches before a run, not in an on-line manner, as we envision for the PROMETHEUS environment. Nevertheless, the system has produced a number of novel reaction pathways that have appeared in the chemistry literature.

Another example is Mitchell et al.'s (1997) DAVICCAND, which was designed to discover quantitative relations in metallurgy. This system encourages the user to actively direct the search process and provides explicit control points where he can influence choices. In particular, the user formulates a problem by specifying the dependent variable the laws should predict, the region of the space to consider, and the independent variables to use when looking for numeric laws. The user can also manipulate the data by selecting which points to treat as outliers. DAVICCAND presents its results in terms of graphical displays and functional forms that are familiar to metallurgists, and the system has produced knowledge published in their literature.

The research that appears closest to our own comes from Mahidadia and Compton (2001), who report an integrated environment for the development and revision of qualitative causal models. Their system provides a graphical interface for model construction and visualization that maps well onto models in their target domain, neuroendocrinology. The JUSTAID system starts with an initial model provided by the user and, using experimental data about the effects of independent variables on dependent measures, recommends changes to this model in terms of link additions and deletions, which the user must approve before they are implemented. The main functional difference between JUSTAID and PROMETHEUS are that the former supports qualitative models, stated as signed links between continuous variables, whereas the latter deals with quantitative process models. Their underlying algorithms also differ, but these are far less visible to users than the model formalism and interface.

[7] A number of commercial environments for knowledge discovery also support user interaction, but, besides focusing on business applications, these emphasize decisions about how to preprocess the data and which algorithm to run on them.

## 9 Directions for Future Research

Although PROMETHEUS breaks new ground in computer-assisted modeling and discovery, we must still extend it along a number of dimensions before it becomes a robust tool for practicing scientists. One limitation is that the current framework only supports models at one level of description, which means that it is most appropriate for situations that involve relatively few variables and processes. A natural response is to expand the modeling language to incorporate the notion of subsystems that characterize components of the overall model. For example, an ecosystem model might include one subsystem for water-related processes and another for sunlight-related processes. This "decompositional" approach would let users hide information and help them manage more complex models by letting them focus both their own attention and that of PROMETHEUS' revision module on one subsystem at a time.

Other extensions would augment the background knowledge available to the environment, which is currently limited to a taxonomy of variables that is linked to a set of generic processes. Future versions of PROMETHEUS should incorporate dimensional information about classes of variables, which would give the system enough knowledge to check models more carefully for correctness and convert units across processes that use different measures. The system should also support a taxonomy of processes to provide the user with more flexibility to direct model revision. For instance, such a taxonomy might include generic processes like 'growth' at higher levels that specify only qualitative proportions between variables, whereas processes at lower levels would encode specialized types like 'exponential growth' that give the forms of numeric equations. Such a hierarchy would let users identify either the abstract processes or the more concrete ones as candidates for revision. More generally, the environment should also support the creation and revision of qualitative models, which are especially appropriate for domains with limited data.

The current implementation of PROMETHEUS relies on a single revision algorithm, but this is certainly not a logical necessity. In future versions, we plan to incorporate other discovery algorithms that would broaden the methods available for model revision, which in turn should make this facility more robust and effective. Additionally, we recognize that scientists will need to manually edit the models that the program produces (e.g., to account for minor discrepancies as discussed at the end of Section 5). In response, we should include tools that ease parameter tuning and sensitivity analysis, which will help the scientist better understand the interrelationships within the model. We should also extend the PROMETHEUS environment to move beyond model revision to support the induction of process models from generic components and data, as we have described elsewhere (Langley et al., 2003).

Finally we plan to modify PROMETHEUS' interface based on the results of our user study. Currently, users instantiate processes by selecting a generic process from a list and specifying the variable bindings and parameter values in a dialog window. Ideally, this activity should take on a more visual character. Research on educational modeling tools such as IQON (Miller et al., 1993) and VMODEL (Bredeweg and Forbus, 2003) suggests possible improvements in this direction. PROMETHEUS also needs tools that help the user understand the deep nature of the relationships among variables. Although the program can produce simulated trajectories, the user must engage in tedious parameter tuning to explore the behavior of a particular model structure. To enable more meaningful exploration of the structure, the environment could create a qualitative version of the model and use a tool like VISIGARP (Salles, 2003) to let the user see the effects of large-scale changes without worrying about parametric details. Such additions will bring the environment closer to a flexible and robust tool that domain scientists would readily adopt and use.

## 10 Concluding Remarks

In this paper, we presented a new framework for modeling and discovering scientific knowledge, along with PROMETHEUS, an interactive environment that implements this approach. The environment includes a language for specifying quantitative models in which the notion of process plays a central role. This formalism takes advantage of traditional scientific notations like algebraic and differential equations, but provides additional structure to aid in presenting and revising models. We illustrated the process modeling language with examples from three domains, which also clarified the interactive features of PROMETHEUS. These include options for visualizing the causal structure of process models, for simulating these models to generate predictions, for visualizing the resulting behavior of models, and for semi-automatically revising models in response to observations. The latter facility lets the user specify the portions of a model to revise and indicate additional processes, taken from a library of generic background knowledge, that the system should consider.

We evaluated this approach to model revision on two domains that concerned interactions within an ecosystem and one that involved gene regulation. Using a simple predator–prey ecosystem, we demonstrated that PROMETHEUS can produce revised models that yield improvements to both the qualitative shape and quantitative error score with respect to the original model. Application to the Ross Sea ecosystem indicated that the environment's revision capabilities scale up to represent more complicated interactions, whereas revision of a photosynthesis-regulation model supported the generality of the approach. In addition, ecologists who worked with PROMETHEUS were interested in the revision module and found its output understandable.

Although our development of PROMETHEUS is still in its early stages, we believe the environment makes important contributions to simulation languages, to human–computer interaction, and to computational scientific discovery. Our initial results with the system have been encouraging and, despite the room that remains for extensions and improvements, we feel that they demonstrate the promise of such an interactive framework for computer-assisted construction and use of scientific models.

## Acknowledgements

The research reported in this paper was supported in part by NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation, in part by Grant NCC 2-1220 from NASA Ames Research Center, and in part by Grant No. IIS-0326059 from the National Science Foundation. We thank Nima Asgharbeygi and Xumei Marker for their early work on the environment's component algorithms, along with Sašo Džeroski, Ljupčo Todorovski, Kazumi Saito, and Daniel Shapiro for discussions that led to many of the ideas in this paper. We also thank Kevin Arrigo and Stuart Borrett for assistance with the ecological aspects of the research.

## References

Arrigo, K.R., Worthen, D.L., Robinson, D.H., 2003. A coupled ocean–ecosystem model of the Ross Sea: 2. Iron regulation of phytoplankton taxonomic variability and primary production. Journal of Geophysical Research 108 (C7), 3231.

Asgharbeygi, N., Langley, P., Bay, S., 2006. Inductive revision of quantitative process models. Ecological Modeling 194, 70–79.

Bredeweg, B., Forbus, K.D., 2003. Qualitative modeling in education. AI Magazine 24, 35–46.

Chown, E., Dietterich, T.G., 2000. A divide and conquer approach to learning from prior knowledge, in: Proceedings of the Seventeenth International Conference on Machine Learning. Morgan Kaufmann, San Francisco, CA, 143–150.

Easley, M., Bradley, E., 1999. Generalized physical networks for automated model building, in: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence. Morgan Kaufmann, 1047–1053.

Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., 1996. From data mining to knowledge discovery in databases. AI Magazine 17, 37–54.

Forbus, K.D., 1984. Qualitative process theory. Artificial Intelligence 24, 85–168.

Forbus, K.D., Falkenhainer, B., 1990. Self-explanatory simulations: An integration of qualitative and quantitative knowledge. in: Proceedings of the Eigth National Conference on Artificial Intelligence. AAAI Press, 380–387.

Jost, C., Ellner, S., 2000. Testing for predator dependence in predator–prey dynamics: A nonparametric approach. Proceedings of the Royal Society of London: Biological Sciences 267, 1611–1620.

Keller, R.M., 1995. An intelligent visual programming environment for scientific modeling. Science Information Systems Newsletter 35.

Labiosa, R., Arrigo, K., Grossman, A., Reddy, T.E., Shrager, J., 2003. Diurnal variations in pathways of photosynthetic carbon fixation in a freshwater cyanobacterium, presented at European Geophysical Society Meeting. Nice, France.

Langley, P., 2000. The computational support of scientific discovery. International Journal of Human–Computer Studies 53, 393–410.

Langley, P., Shiran, O., Shrager, J., Todorovski, L., Pohorille, A., 2006. Constructing explanatory process models from biological data and knowledge. Artificial Intelligence in Medicine 37, 191–201.

Langley, P., George, D., Bay, S., Saito, K., 2003. Robust induction of process models from time-series data, in: Proceedings of the Twentieth International Conference on Machine Learning. Washington, DC, 432–439.

Langley, P., Simon, H.A., Bradshaw, G.L., Żytkow, J.M., 1987. Scientific discovery: Computational explorations of the creative processes. MIT Press, Cambridge, MA.

Levenberg, K., 1944. A method for the solution of certain problems in least squares. Quarterly of Applied Mathematics 2, 164–168.

Mahidadia, A., Compton, P., 2001. Assisting model discovery in neuroendocrinology, in: Proceedings of the Fourth International Conference on Discovery Science. Springer, 214–227.

Marquardt, D., 1963. An algorithm for least-squares estimation of nonlinear parameters. SIAM Journal of Applied Mathematics 11, 431–441.

Miller, R., Ogborn, J., Briggs, J., Brough, D., Bliss, J., Boohan, R., Brosnan, T., Mellar, H., Sakonidis, B., 1993. Educational tools for computational modelling. Computers in Education 21, 205–261.

Mitchell, F., Sleeman, D., Duffy, J.A., Ingram, M.D., Young, R.W., 1997. Optical basicity of metallurgical slags: A new computer-based system for data visualisation and analysis. Ironmaking and Steelmaking 24, 306–320.

Ourston, D., Mooney, R., 1990. Changing the rules: A comprehensive approach to theory refinement, in: Proceedings of the Eighth National Conference on Artificial Intelligence. AAAI Press, Boston, MA, 815–820.

Richmond, B., Peterson, S., Vescuso, P., 1987. An academic user's guide to STELLA. High Performance Systems, Lyme, NH.

Saito, K., Langley, P., Grenager, T., Potter, C., Torregrosa, A., Klooster, S.A., 2001. Computational revision of quantitative scientific models, in: Proceedings of the Fourth International Conference on Discovery Science. Springer, 336–349.

Shneiderman, B., 2000. Creating creativity: User interfaces for supporting innovation. ACM Transactions of Computer-Human Interaction 7, 114–138.

Salles, P., Bredeweg, B., 2003. Qualitative reasoning about population and community ecology. AI Magazine 24, 77–90.

The MathWorks, Inc., 1997. SIMULINK user's guide: Dynamic system simulation for MATLAB. Natick, MA.

Todorovski, L., 2003. Using domain knowledge for automated modeling of dynamic systems with equation discovery. Doctoral Dissertation, Faculty of Computer and Information Science, University of Ljubljana, Slovenia.

Todorovski, L., Džeroski, S., 1997. Declarative bias in equation discovery, in: Proceedings of the Fourteenth International Conference on Machine Learning. Morgan Kaufmann, Nashville, TN, 376–384.

Todorovski, L., Džeroski, S., 2001. Theory revision in equation discovery, in: Proceedings of the Fourth International Conference on Discovery Science. Springer, Washington, DC, 389–400.

Towell, G., 1991. Symbolic knowledge and neural networks: Insertion, refinement, and extraction. Doctoral dissertation Computer Sciences Department, University of Wisconsin, Madison, WI.

Valdés-Pérez, R.E., 1995. Machine discovery in chemistry: New results, Artificial Intelligence 74, 191–201.

Washio, T., Motoda, H., 1998. Discovering admissible simultaneous equations of large scale systems, in: Proceedings of the Fifteenth National Conference on Artificial Intelligence. AAAI Press, Madison, WI, 189–196.