Advances in Cognitive Systems 4 (2016) 207-226 Submitted 8/2015; published 6/2016

Heuristic Adaptation of Scientific Process Models

Adam Arvay AARV914@AUCKLANDUNI.AC.NZ
Pat Langley PATRICK.W.LANGLEY@GMAIL.COM
Department of Computer Science, University of Auckland, Private Bag 92019, Auckland 1142, NZ

Abstract

Scientific models are seldom constructed from scratch; more often they are adapted from some existing models. In this paper, we present a computational approach to this adaptation task in the context of quantitative process models. We review the paradigm of inductive process modeling and discuss RPM, a recent system that operates on this problem. After this, we describe APM, a new system that adapts a process model to a new setting by revising its parameters or altering its component processes. Next we report experiments that demonstrate the system's basic abilities and compare its efficiency relative to using RPM. We conclude by discussing other research on model revision and outlining plans for additional work.

1. Background and Motivation

Research on computational scientific discovery (Shrager & Langley, 1990; Džeroski, Langley, & Todorovski, 2007) addresses the construction of laws and models in established scientific formalisms. Much work on this topic has dealt with finding empirical relations that describe regularities in data, such as those appearing early in the stages of a field's development. There has been much less research on the construction of explanatory models that move beyond the data to account for observations at a deeper level. Work in this area incorporates structured representations and multi-step reasoning over these structures, making the task especially relevant to the cognitive systems community.

In this paper, we focus on the problem of *inductive process modeling* (Langley, Sanchez, Todorovski, & Džeroski, 2002). Here one is provided with multivariate time series for some dynamic system and background knowledge about the types of processes that can occur in the domain. The goal is to generate a quantitative process model, including numerical parameters, that reproduces the observed trajectories and that predicts new values accurately. Such a model compiles into a set of linked differential equations, but it also accounts for the data in terms of unobserved processes. This distinguishes research in the area from work on differential equation discovery such as that by Džeroski and Todorovski (2008), which describes but does not explain them in deeper terms.

However, in many cases, scientists are less concerned with creating a model from the ground up than with adapting an existing model to a new setting. This may occur when they believe the system under study has changed, so that the model no longer fits observations as well as those for which they devised it. Or they may have developed the model to explain data from one area and

find that it does not fare as well on data for an adjacent area. In either case, they may need to revise the model's parameters to address quantitative changes, or even need to alter the model's structure by removing, adding, or replacing some of its component processes.

In the sections that follow, we describe one approach to the task of adapting a process model to explain data in a new setting. We start by reviewing RPM, a recently developed system for process model induction that is both more reliable and more efficient than earlier approaches. After this, we describe APM, a new system for model adaptation that builds on the ideas that underlie RPM's successes. Next we report empirical studies designed to show that APM operates as intended and that it offers efficiency gains over inducing a model from scratch. Finally, we discuss previous work on model revision and propose directions for future research.

2. Review of the RPM System

In recent research, we have reported a new approach to inductive process modeling and its implementation in the RPM system (Langley & Arvay, 2015). The new framework builds on earlier ones (Borrett, Bridewell, Langley, & Arrigo, 2007) but also introduces important new ideas about representation and processing. In this section, we review these two aspects of RPM in turn, along with some experimental results.

2.1 Representation in RPM

Like its predecessors, RPM organizes differential equation models into distinct *processes*. These identify aspects of the equations that must stand or fall together. For example, ecosystem models include processes such as predation, grazing, growth, loss, and nutrient absorption. However, the system differs from earlier ones by making four key assumptions:

- All processes concern changes over time and effect these changes at a specific *rate*. For instance, a chemical reaction describes interactions among a set of substances, but its rate of operation can vary over time.
- Each process has one or more associated *derivatives* that are proportional to its rate. Some variables are *inputs* to a process, which it consumes and thus have negative coefficients, while others are *outputs*, which a process produces and thus have positive coefficients.
- The rate of each process is determined by a *parameter-free* algebraic expression. RPM assumes that rates are always positive and inherently unobservable, so it can adopt any measurement scale it likes, avoiding the need for coefficients.
- If a variable appears in the rate expression for a process, then it must also appear as a derivative associated with that process.

Along with the standard supposition that the effects of different processes are additive, these postulates mean that one can compile a process model into a set of differential equations that are linear combinations of algebraic rate expressions. When joined with a fifth assumption, that all variables are observed on each time step, this suggests a novel approach to inducing process models that it both efficient and robust.

Table 1. (a) A three-process model for a system involving the predator *Nasutum* and the prey *Aurelia* that illustrates RPM's assumptions. Each process has a rate determined by an algebraic expression and includes one or more derivatives that are proportional to this rate. (b) The differential equation model into which the process model compiles.

(a)
```
exponential_change[aurelia]
  rate        r = aurelia
  equations   d[aurelia] = 0.75 × r

exponential_change[nasutum]
  rate        r = nasutum
  equations   d[nasutum] = -0.57 × r

holling_predation[nasutum, aurelia]
  rate        r = nasutum × aurelia
  equations   d[nasutum] = 0.0024 × r
              d[aurelia] = -0.011 × r
```

(b)
```
d[aurelia] = 0.75 × aurelia + -0.011 × nasutum × aurelia
d[nasutum] = 0.0024 × nasutum × aurelia + -0.57 × nasutum
```

Table 1 presents a simple predator-prey model that illustrates these ideas. Each process has an associated rate expression, one specifying that its rate equals the product of two variables and the others stating that its rate equals a single variable. Each process also has one or more associated derivatives that are proportional to its rate, with parameters detailing this dependence and with $d[x]$ referring to the first derivative of x with respect to time. The formalism requires that each process include both elements, with complex functions restricted to the rate expression. These assumptions constrain the space of possible process models, although they still allow many such structures. The table also shows the corresponding set of differential equations in the constrained form.

Figure 1 shows the process diagram representation of three process predator prey model. Ellipses enclose variables and rectangles indicate processes. The direction of the links between variables and processes show a positive or negative influence on the variable's derivative. The exponential_change process for growth has a positive influence over the derivative of *Aurelia*. Neither the rate equation nor specific coefficient values are shown in the diagram. Holling_predation both decreases aurelia and increases *Nasutum*. The number of arrows connected to a process indicates the number of equations in which that process appears. Similarly, the number of arrows connected to a variable indicates how many processes appear in its differential equation.

RPM's formalism incorporates key ideas from Forbus' (1984) Qualitative Process Theory, in which rates also played a key role. Our algebraic rate expressions correspond, in his notation, to a set of indirect influences associated with a qualitative process. Similarly, each equation in our framework that relates a derivative to a rate map, in Forbus' framework, onto a direct influence. Qualitative Process Theory allows multiple rates per process while RPM allows only one, and of course our models are quantitative rather than qualitative, but otherwise they have many common features, including a central concern with behavior over time.

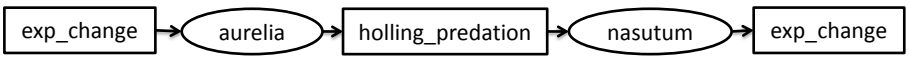

Figure 1. A process diagram for the predator-prey ecosystem model in Table 1. Rectangles stand for processes and ellipses denote variables. Directed arrows indicate whether the influence of a process on a variable's derivative is positive (incoming) or negative (outgoing).

The system also encodes background knowledge about *generic* processes, which take a similar but more abstract form. Each generic process specifies one or more variables that it relates, the structure of the algebraic expression that determines its rate, and one or more derivatives that are proportional to this rate. A generic process does not mention either specific variables or parameter values, but it can indicate constraints. For instance, it may specify the type of each variable (say *predator* or *nutrient*) and it may place bounds on parameter values (say positive or negative). Each process also has a type, such as *predation* or *nutrient absorption*, with alternatives differing in their functional forms. Generic processes play the same role as building blocks for model induction as in earlier work on this topic (Bridewell et al., 2008), as well as in research in automated construction of qualitative models (Crawford, Farquhar, & Kuipers, 1990).

2.2 Inducing Rate-Based Process Models

Earlier approaches to inductive process modeling generated many alternative model structures and then fit their parameters to observations in order to evaluate them. Parameter estimation involved repeated simulation of each model structure with different values and calculation of an error to drive gradient descent search, with random restarts to reduce the chances of finding local optima. This scheme was computationally expensive, scaled poorly to complex models, and even so did not always find good parameterizations.

RPM organizes induction in a very different manner, carrying out heuristic rather than exhaustive search through the space of model structures by finding the equation for one derivative at a time. The system starts by instantiating its generic processes in all ways that are consistent with their constraints to produce a set of process instances (e.g., that organism A preys on organism B). The system also calculates the rates for each process instance on each time step, which is possible because rate expressions contain no parameters and because all variables are observed.

Next RPM selects a derivative on which to focus, retrieves all process instances in which it appears, and attempts to induce a differential equation that predicts the derivatives, which it estimates using a simple 'center difference' method (Mathews & Fink 2004) on each time step. Because the empirical derivative must be a linear combination of process rates, and because the latter are known, RPM invokes multiple linear regression for this purpose. However, the system does not know in advance which process rates are relevant, so it first considers all equations that are functions of individual process rates. If none of these is accurate enough (as reflected by r^2), then it considers equations that include all pairs of process rates, continuing until it finds an acceptable equation or it exceeds some user specified maximum number of processes.

Once RPM has found an equation for the first derivative term, it selects another one and repeats the process. The system takes the partial model it has already constructed into account at later stages.

For example, it selects for attention the derivative that appears in the most processes incorporated to date. Moreover, it requires that, if a derivative d appears in a process p used in an earlier equation, then the equation for d must include p. RPM also uses constraints on process types and their parameter ranges to reject some candidates, further reducing search through the space of model structures and making induction tractable.

2.3 Experimental Studies of RPM

In addition to describing RPM, Langley and Arvay (2015) reported experimental studies of the system's behavior. They showed that it constructs an accurate and plausible model from published observations on a simple predator-prey ecosystem, similar in form to those found by earlier programs. They also used synthetic data to demonstrate that RPM can ignore irrelevant processes, handle noisy data when aided by standard smoothing techniques, and scale well enough to induce process models with 20 variables. These capabilities appear to follow directly from its use of heuristic, rather than exhaustive, search through the space of model structures and its reliance on linear regression, rather than gradient descent with random restarts, to estimate equation parameters.

The authors also reported scaling studies in which they varied systematically the number of processes provided as background knowledge and the number of variables in the target model. The processing time needed for model construction grew slowly in both cases, with an increase in candidate processes having greater cost than growth in model complexity. Both curves appeared polynomial, which was consistent with a worst-case analysis of the approach. In addition, they compared RPM's behavior to that of SC-IPM (Bridewell & Langley, 2010), an earlier system for inductive process modeling that uses combinatorial search and gradient descent. They found that, on synthetic data for a three-variable predator-prey task, RPM found accurate models far more reliably than its predecessor and, at worst, ran 83,000 times more rapidly. However, they designed the system to construct process models from scratch, rather than to adapt an existing model to a new setting. This task of adaptation suggests even more efficient ways to approach process model induction that build on RPM's insights but also extend them, as we discuss in the section that follows.

3. An Approach to Process Model Adaptation

Adapting an existing process model to explain and describe new time series should be less computationally intensive than constructing a model from scratch. Rate-based process models of the type just described are particularly well suited to revision because they consist of equations that one can evaluate independently. This separation of model components makes it straightforward to determine which equations to alter and which ones to retain without changes. To explore this prospect we have developed APM, for *Adaptive Process Modeler*, a system that revises an existing process model to explain newly observed data when needed. The program is implemented in LISP and operates in three stages, which we now describe in turn.

3.1 Detecting Anomalous Derivatives

The first step in APM's model adaptation is to determine whether any equations require revision and, if so, which ones. This stage takes as inputs an existing base model, new time-series data to test it against, and criteria about acceptable deviations. The user must specify the correspondence

Table 2. (a) Quantitative process model for a six-variable predator-prey ecosystem and (b) the set of differential equations into which the model compiles.

(a)

```
exponential_change[x1]
  rate         r = x1
  equations    d[x1] = 1.7 × r

holling_predation[x1, x2]
  rate         r = x1 × x2
  equations    d[x1] = -0.8 × r
               d[x2] = 1.3 × r

holling_predation[x2, x3]
  rate         r = x2 × x3
  equations    d[x2] = -1.4 × r
               d[x3] = 0.8 × r

holling_predation[x3, x4]
  rate         r = x3 × x4
  equations    d[x3] = -0.9 × r
               d[x4] = 1.1 × r

holling_predation[x4, x5]
  rate         r = x4 × x5
  equations    d[x4] = -0.8 × r
               d[x5] = 0.8 × r

holling_predation[x5, x6]
  rate         r = x5 × x6
  equations    d[x5] = -1.0 × r
               d[x6] = 0.9 × r

exponential_change[x6]
  rate         r = x6
  equations    d[x6] = -1.1 × r
```

(b)

```
(1) d[x1] = 1.7 × x1 - 0.8 × x1 × x2
(2) d[x2] = 1.3 × x1 × x2 - 1.4 × x2 × x3
(3) d[x3] = 0.8 × x2 × x3 - 0.9 × x3 × x4
(4) d[x4] = 1.1 × x3 × x4 - 0.8 × x4 × x5
(5) d[x5] = 0.8 × x4 × x5 - 1.0 × x5 × x6
(6) d[x6] = 0.9 × x5 × x6 - 1.1 × x6
```

between variables in the base model and new data set. The base model comprises a set of rate-based processes that maps onto a set of linear differential equations. This model predicts the derivative for each dependent variable on each time step, and these predictions are available for comparison to the corresponding 'observed' derivatives for each variable.

APM uses the r^2 statistic, a measure of the variance explained, as its criterion for detecting anomalous behaviour. We assume the system has access to the base model's r^2 score for each variable on the initial data set. The user specifies a fraction (e.g., 0.8 or 0.9) of the prior score that would be acceptable for the model's fit to the new data set. The system tests the base model on the new data, calculates an r^2 score for each equation, and computes the ratio between the new and prior scores. If this ratio falls below the user-specified threshold, APM marks the equation as a target for revision. Based on this analysis, the anomaly detection module outputs two sets of equations, one containing elements that need revision and another whose equations need not be altered.

As an example, suppose we have the base process model in Table 2 (a), which compiles into the six differential equation in Table 2 (b) and which produces the trajectories in Figure 2. Further suppose that the user decides to reuse this model to explain the new trajectories in Figure 3, which are consistent with the process model and equations in Table 3. Note that three equations are identical in the two models, but the equation for $d[x2]$ differs in its parameters, while those for $d[x1]$ and

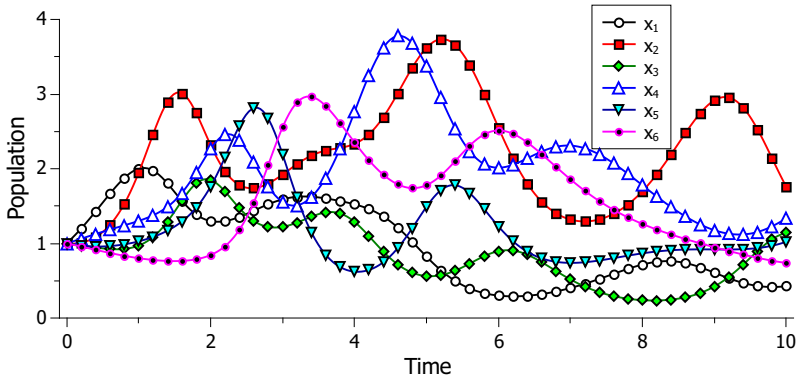

Figure 2. Trajectories for the six-variable predator-prey explained by the base model in Table 2.

$d[x3]$ differ in their structure. In this example, we can see that the equations for $d[x4]$, $d[x5]$ and $d[x6]$ are identical in both models. However, the trajectories for each equation in both models are very different. It may seem counter intuitive for identical equations with the same intital conditions to produce different trajectories, but remember that they are part of a fully connected system, so changes to any variable can influence all of the others. APM does not have any details about the model that generated the new data; it knows only the initial data set, the base model, and new data that it has been asked to explain.

Figure 4 plots the derivatives estimated from observed values against those predicted by the base model equations on the new data set. APM uses the observed values of variables to calculate process rates, which the equations then combine to generate predicted derivatives on each time step. A model that makes accurate predictions will have points that fall along the diagonal line, as with the equation for $d[x4]$. The agreement between observed and predicted derivatives is especially poor for $d[x1]$'s equation, reflecting a low r^2 score of -4.8. Suppose that, in this case, the user has decided that a revision threshold of 0.8 is appropriate and the ratio of r^2 scores on the old and new data sets exceed this threshold for equations 4, 5, and 6. APM retains those equations for the final model with no changes. However, suppose further that equations 1, 2 and 3 have ratios that fall below 0.8. This leads the system to mark them for revision during later stages of processing.

3.2 Revising Equation Parameters

Once APM has identified which equations, if any, require revision, it attempts to reestimate their parameters. Recall that each equation is stated as a linear combination of process rates, with the latter being parameter-free algebraic combinations of observed variables. Inputs to this module include a set of equations to be revised, the set of derivatives estimated from observed trajectories, and the generic forms of processes that appear in the base model.

The generic processes are necessary because they contain information about constraints on parameter values, such as whether they must be positive or negative. These constraints reduce the chances that regression analysis will find parameters that fit the data but do not generalize. The outputs from the parameter revision module are two sets of equations. One contains expressions with

Table 3. (a) Quantitative process model for a six-variable ecosystem that is the target for revision and (b) the set of differential equations into which the model translates. Differences from the base model in Table 2 are indicated in boldface.

(a)

exponential_change[x1]		holling_predation[x3, x4]
rate	r = x1	rate r = x3 × x4
equations	d[x1] = 1.7 × r	equations d[x3] = −0.9 × r
		d[x4] = 1.1 × r
holling_predation[x1, x2]		
rate	r = x1 × x2	holling_predation[x4, x5]
equations	d[x1] = −0.8 × r	rate r = x4 × x5
	d[x2] = **0.25** × r	equations d[x4] = −0.8 × r
		d[x5] = 0.8 × r
holling_predation[x1, x3]		holling_predation[x5, x6]
rate	**r = x1 × x3**	rate r = x5 × x6
equations	**d[x1] = −0.8 × r**	equations d[x5] = −1.0 × r
	d[x3] = 1.1 × r	d[x6] = 0.9 × r
holling_predation[x2, x3]		
rate	r = x2 × x3	exponential_change[x6]
equations	d[x2] = **−0.7** × r	rate r = x6
	d[x3] = 0.8 × r	equations d[x6] = −1.1 × r

(b)

(1) d[x1] = 1.7 × x1 − 0.8 × x1 × x2 **− 0.8 × x1 × x3**
(2) d[x2] = **0.25** × x1 × x2 − **0.7** × x2 × x3
(3) d[x3] = 0.8 × x2 × x3 − 0.9 × x3 × x4 + **1.1 × x1 × x3**
(4) d[x4] = 1.1 × x3 × x4 − 0.8 × x4 × x5
(5) d[x5] = 0.8 × x4 × x5 − 1.0 × x5 × x6
(6) d[x6] = 0.9 × x5 × x6 − 1.1 × x6

new parameter values that exceed the threshold for r^2 ratio between the initial and new data. A second set, possibly empty, contains derivatives for which APM could not find acceptable parameters using the original equation structure.

Recall that, in our example, APM has marked equations 1, 2, and 3, along with their associated derivatives, as requiring adaptation because their r^2 ratios were below the 0.8 threshold. In response, the parameter revision module invokes multivariate regression on each derivative to determine new parameter values, respecting the constraints in the generic processes. After this step, equations 4, 5, 6, and 2 are in the final model, while equations 1 and 3 remain in the revision set for future processing. In this case, APM cannot find acceptable equations for $d[x1]$ and $d[x3]$ because, in the true model (which it is attempting to induce), different structures determine these derivatives. As a result, parameter revision is insufficient and the system continues to the next stage.

3.3 Adapting Equation Structure

If APM decides that anomalous variables cannot be handled by parameter revision, it resorts to structural adaptation, which adds or removes processes from a model. This stage takes as inputs the results of parameter revision and the known generic processes. The latter may include processes

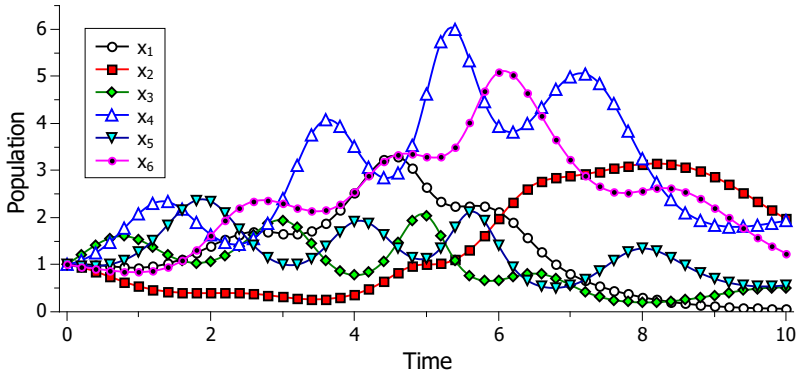

Figure 3. Trajectories for the six-variable target predator-prey model in Table 3.

that did not appear in the base model but that are plausible candidates for the domain. This module constructs a set of process instances that include the anomalous derivative terms. It also examines the processes that appear in the equations for other derivatives to determine if any of them must appear in the equations it is about to construct. Process instances can only be added or removed from anomalous variable equations, as making changes to a nonanomalous equation would alter the r^2 value that was previously declared to be acceptable.

APM begins structure revision with the variable that has the most processes already determined. This keeps the new equation consistent with the existing model and reduces the size of the search space. Like RPM, the system carries out breadth-first search, ordered by complexity, for new equation structures. The starting structure contains only the process rates required by their appearance in other equations. The module creates more complex equations by adding rate terms until it reaches a maximum number of processes. Search terminates when an equation at a given level exceeds the threshold for r^2 ratios or it reaches a maximum number of processes. In cases where multiple equations at a given complexity level exceed the threshold, APM returns the best-scoring candidate. The final output is the set of equations found during search plus those produced by the previous stage.

Let us return to our example, with structural revision continuing where the parameter estimation procedure left off. Recall that APM had altered the parameters for the $d[x2]$ equation, but that this did not suffice for $d[x1]$ or $d[x3]$. Because the system must only find equations for these derivatives, it generates only those process instances that contain them as dependent variables. In response, APM carries out a structural search for equations that explain their estimated values as a linear combination of process rates. Rather than starting from scratch, as in RPM, this begins with a partial model that includes the equations from earlier stages.

The process diagram in Figure 5 shows the base model with the additional process in dashed outlines. There are no new arrows connecting to $d[x2]$, $d[x4]$, $d[x5]$, or $d[x6]$, meaning that the structure for those equations is unchanged. The new predation process is connected to $x1$ and $x3$, which means it appears in the equations for $d[x1]$ and $d[x3]$. Furthermore the direction of the arrows indicate that the process has a negative coefficient for $d[x1]$ and a positive one for $d[x3]$. The specific values of the coefficients are found during parameter estimation.

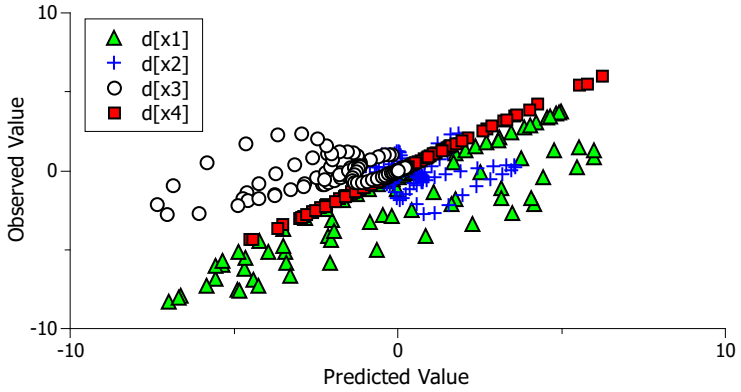

Figure 4. Observed vs. predicted derivatives for four variables from the model in Table 2. Acceptable equations have points that fall along the diagonal line, whereas ones that require revision are widely scattered.

The rate-based process framework assumes that any variable appearing in a rate expression must also be influenced by that rate. Thus, if the rate expression $x2 \times x3$ appears, then it must appear in equations for both $d[x2]$ and $d[x3]$. Examination of equations 2, 4, 5, and 6 from the base model in Table 2 reveal that equation 2 contains the rate expression $x1 \times x2$, which means that same rate expression must appear in the revised version of equation 1. In addition, the appearance of the term $x2 \times x3$ in equation 2 and the appearance of $x3 \times x4$ in equation 4 determine that both $x2 \times x3$ and $x3 \times x4$ must appear in equation 3. APM begins from equation 3 rather than equation 1 because it has two required rate terms, rather than only one. At the end of its structure search, the system returns the model in Table 3. This carries over equations 4, 5, and 6 from the base model, but equation 2 has altered parameters, whereas equations 1 and 3 have entirely new structures.

In summary, model adaptation in APM operates in three stages: anomaly detection, parameter revision, and structure adaptation. Each derivative is tested independently, which simplifies anomaly detection and lets the system identify which parts of the base model it need not change. Parameter revision involves the reestimation of rate coefficients using multiple linear regression with known terms. Structure revision requires more effort, but it takes existing equations and their constituent processes into account to determine the order in which to incorporate derivatives and to reduce the number of candidate structures considered. This approach to model adaptation builds on the strengths of the rate-based process framework discussed by Langley and Arvay (2015) to adapt existing models to new data in an efficient manner.

4. Experimental Evaluation

We have designed APM to take an existing base model and adapt it to explain new data. The system attempts to detect when equations should change, revise their parameters if possible, and to alter their structure if necessary. In this section, we report runs on synthetic data that demonstrate APM's basic abilities for parameter and structure revision across four distinct types of revision scenarios. We report CPU times for each scenario compared against inducing a process model from scratch.

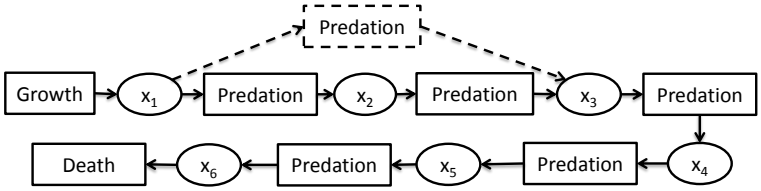

Figure 5. Process diagram for the six-variable target predator-prey model in Table 3.

We also demonstrate generality by evaluating APM on a more complex aquatic ecosystem domain. In addition, we present the results of a scaling study that compares APM's efficiency as the number of variables in the model increases.

4.1 Basic Functionality

Our initial evaluation aimed to demonstrate that APM has the intended ability to identify anomalies and revise the base model in response. We used six-variable predator-prey models similar to those in Table 2 to generate synthetic trajectories for testing purposes. Langley and Arvay (2015) report that RPM is robust to reasonable amounts (ten percent) of noise, so we did not add random variation to these data. We tuned parameter values to produce stable trajectories over the observed time frame. We provided APM both with the handcrafted base model and with appropriate generic processes that specified the algebraic forms for rate terms and constraints on parameters. We set the anomaly detection threshold to 0.8, which means that, to be acceptable, the model's r^2 on the new data must be no less than 0.8 of that on the original data.

In the first study we evaluated APM's ability to revise model parameters. We altered coefficients in the first two equations in the base model and used the result to generate new trajectories. In this case, the system calculated r^2 values of 0.56, -2.27, 0.99, 0.99, 0.99, and 0.99 on the new data. The corresponding r^2 scores on the initial data were all 0.99, which gave ratios of 0.57, -2.29, 1.0, 1.0, 1.0, and 1.0. The first two fell below the user-specified 0.8 threshold, so APM reestimated the parameters for these equations, which produced r^2 scores of 0.99 in each case. In response, the system incorporated them into the model it returned without resorting to structure revision.

Our second study evaluated APM's ability to adapt model structure. We created the target model by modifying the base model structure, adding one process that affects variables $d[x1]$ and $d[x3]$, then used the result to generate new trajectories. APM calculated r^2 values of -3.3 and -5.1 for $d[x1]$ and $d[x3]$, respectively, with all other values above the revision threshold. In repsonse APM first attempted to revise the parameters for these equations, resulting in new r^2 values of 0.71 and 0.21 for $d[x1]$ and $d[x3]$. The resulting ratios fell below the revision threshold, so the system invoked its structure revision module. This produced new equations for $d[x1]$ and $d[x3]$ that included the correct new process and corresponding r^2 scores of 0.99. This result showed that APM can adapt model structure when parameter revision fails.

Our third run aimed to demonstrate APM's ability to handle cases that require both parametric and structural changes. Here we provided the system with the trajectories in Figure 3, which we generated using the target model in Table 3. In this case, anomaly detection noted that the r^2 values

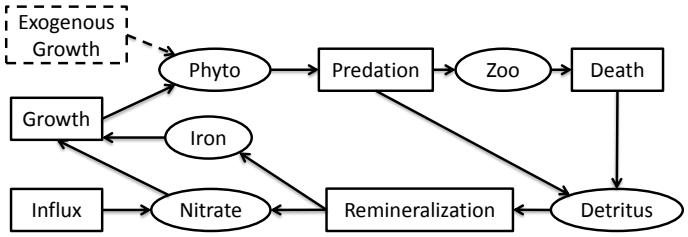

Figure 6. Process diagram of the aquatic ecosystem model. The target model includes an exogenous growth process, indicated by dashed outlining, that influences the phyto variable.

drop from 0.99 for all derivatives to -3.13, 0.22, and -4.84 for $d[x1]$, $d[x2]$ and $d[x3]$, with the others remaining very high. APM behaved as described earlier, first finding that the r^2 ratios for these derivatives are below the 0.8 threshold and then revising parameters for their equations to obtain r^2 values of 0.58, 0.99, and 0.19 for $d[x1]$, $d[x2]$, and $d[x3]$, respectively. The ratios for $d[x1]$ and $d[x3]$ were still below the 0.8 threshold, so the system attempted to adapt the structure of their equations, after incorporating the updated equation for $d[x2]$ into the new model. The structural revision module returned new equations for $d[x1]$ and $d[x3]$ (shown in Table 3), each with 0.99 as their r^2 score. These results demonstrate that APM first revises parameters and then resorts to structural adaptation only for equations that require it.

The fourth evaluated APM's ability to handle data sets that contain new variables. This adaptation task differs from earlier variants in that one must create equations from scratch for the new terms. We constructed a target model by extending the linear food chain of the base model with two new variables, $d[x7]$ and $d[x8]$, including a new process that influences $d[x6]$ and $d[x7]$ connecting them to others. APM began anomaly detection on the variables present in the base model, returning an r^2 of 0.56 for $d[x6]$ and high scores for the other variables. Parameter revision on $d[x6]$ returned an updated score of 0.62, which was still below the revision threshold. APM went on to revise the structure for $d[x6]$, returning a new equation with an r^2 value of 0.99. In this model, $d[x6]$ and $d[x7]$ are influenced by the same process, so once the system found the equation for $d[x6]$, a portion of the equation for $d[x7]$ had also been determined. APM then induced an equation for $d[x7]$ before moving on to $d[x8]$, the final variable. The resulting model included two new equations, one revised equation, and five carried over from the base, all with r^2 scores of 0.99. This study demonstrated APMs ability to add equations for entirely new variables while adapting others as necessary. Each study demonstrated a specific revision ability but any combination of them can occur together, such as different parameter values along with new variables and generic processes.

4.2 Generality of the Approach

In order to demonstrate APM's generality, we repeated the previous experiments in a more complex setting based on an aquatic ecosystem model described by Bridewell et al. (2008). Figure 6 shows the process diagram for the base and target model, whereas Table 4 provides their details. This model involves more interaction among variables than the predator-prey food chain models used in

Table 4. (a) Quantitative process model for the base aquatic ecosystem, with differences between the base and target model are indicated in boldface text. (b) The set of differential equations into which the model translates. Equation 3T shows the parameter values used for the altered equation in the target model.

(a)

Growth[phyto, iron, nitrate]		
rate	r = phyto × iron × nitrate	
equations	d[phyto] = 4.0 × r	
	d[nitrate] = −2.8 × r	
	d[iron] = − 1.15 × r	
predation[phyto, zoo, detritus]		
rate	r = phyto × zoo	
equations	d[phyto] = −1.9 × r	
	d[zoo] = 1.5 × r	
	d[detritus] = 1.0 × r	
death[zoo, detritus]		
rate	r = zoo	
equations	d[zoo] = −0.75 × r	
	d[detritus] = 0.64 × r	
remineralization[detritus, nitrate, iron]		
rate	r = detritus	
equations	d[detritus] = −1.28 × r	
	d[nitrate] = 0.20 × r	
	d[iron] = 0.33 × r	
influx[nitrate]		
rate	r = 1/nitrate	
equations	d[nitrate] = 0.5 × r	
exo_growth[phyto]		
rate	**r = phyto × exo**	
equations	**d[phyto] = 1.2 × r**	

(b)

(1) d[phyto] = 4.0 × phyto × iron × nitrate − 1.9 × phyto × zoo + **1.2 phyto × exo**
(2) d[zoo] = 1.5 × phyto × zoo − 0.75 × zoo
(3) d[nitrate] = −2.8 × phyto × iron × nitrate + 0.2 × detritus + 0.5 × 1/nitrate
(3T) d[nitrate] = **−3.2** × phyto × iron × nitrate + **0.4** × detritus + **0.3** × 1/nitrate
(4) d[iron] = −1.15 × phyto × iron × nitrate + 0.33 × detritus
(5) d[detritus] = 0.64 × zoo + 1.0 × phyto × zoo − 1.28 × detritus

the previous section. Predator and prey species still appear in this ecosystem, but so do several other variables, and some processes have more than one input or output. This model's structure relates five variables and five processes through an extended feedback loop.

The aquatic ecosystem model has phytoplankton as a primary producer that increases via a growth process, absorbing the nutrients iron and nitrate. Zooplankton consumes phytoplankton though a predation process that reduces the phytoplankton population, increases the zooplankton population, and produces detritus. Zooplankton population decreases through a death process that also produces detritus. Remineralization reduces detritus and produces iron and nitrate, while an external source of nitrate also influences nitrate concentration. The feedback loop closes as the

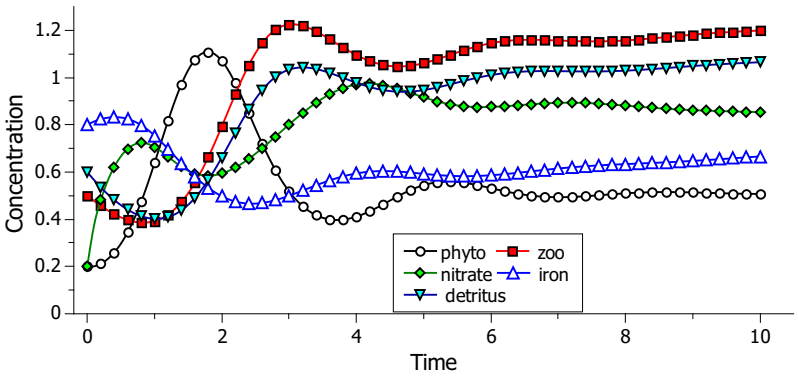

Figure 7. Trajectories of the base aquatic ecosystem model.

growth process consumes the iron and nitrate to produce phytoplankton. We tuned parameter values so that the model tends toward steady-state values for all variables, as shown in Figure 7.

We changed the target model by adjusting the parameters in the nitrate equation and by adding the exogenous growth process that influences only the phyto variable. The rate equation for the exogenous growth process includes the exogenous variable. The purpose of including a new term is to demonstrate APM's ability to handle exogenous variables and to produce very different dynamics in the target data, as we wanted to show that the system identifies and retains appropriate parts of the model despite very different trajectories. As Figure 8 shows, the target model tends to oscillate about an equilibrium value instead of approaching steady state.

We evaluated APM's ability to adapt this more complex model structure starting from the base model in Table 4 (less the exogenous growth process) and given the trajectories in Figure 8. Adaptation proceeded in the same manner as before. APM first checked the fit of each equation in the base model on the new data, determining that the equation for $d[phyto]$ and the equation for $d[nitrate]$ have r^2 values of 0.47 and -1.92 while those for other equations were high. These two values fell below the user specified ratio of 0.8, so the system estimated new parameters for those equations. APM found that the new r^2 values for $d[phyto]$ and $d[nitrate]$ were 0.65 and 0.99, the latter being high enough to be included in the new model. After this, it went on to search for a new structure for $d[phyto]$ in the same manner as the previous study, eventually returning an r^2 of 0.99.

Adaptation in this setting works in the same manner as in the simpler one. More complex models with processes that influence more variables are handled just as easily. The system detects anomalous derivatives and adapts them as necessary, first by altering parameters and then changing structure, if required, by adding new processes or new variables if appropriate. The approach treats each equation independently, so complexity in the overall model structure need not introduce complexity in individual equations, which APM still finds using linear regression. These runs provide evidence that APM can take an existing process model and adapt it as needed to explain new trajectories in a variety of different circumstances. They show that the system supports the basic abilities that we intended, but we are also interested in other aspects of its behavior.

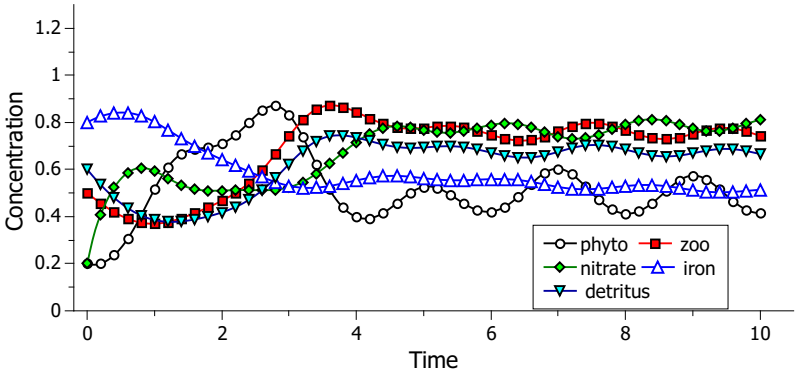

Figure 8. Trajectories for the target aquatic ecosystem model from Table 4.

4.3 Computational Benefits

We argued earlier that adapting an existing process model to new data should be more efficient computationally than constructing a new model from scratch. Our approach to parameter revision, which calls multivariate regression, requires no search at all, so these benefits should be largest when only changes to model coefficients are needed, but there should also be savings when structural changes arise. The reductions for parameter revision are due in part because it sidesteps calculation of unnecessary process rates. APM's use of linear regression to estimate parameters depends on these rates being calculated from observed values on each time step. To this end, the system calculates the process rates that appear in the equations for that model structure. When structural adaptation is necessary, there are still computational savings, as the size of the space is reduced substantially because only a subset of rate terms (ones that mention the derivative) are relevant. This decreases the number of process instances generated, thus decreasing the combinations that can appear in an equation. In addition, APM uses existing equations to constrain structural revision by keeping revised equations consistent with them.

We can demonstrate these computational savings by comparing the CPU time needed for each revision run in Subsection 4.1 with the time to construct a model from scratch using the RPM system. These studies used the six-variable predator-prey model from Table 2 as the base model. In cases where only parameters need to be estimated, APM revised the base model to produce the target model in 1.2 ms, while RPM requires 19.1 ms (averaged across 50 runs) to produce the target model. When structural adaptation is necessary, APM adapts the base model in 5.5 ms while RPM requires 29.4 ms. Similar results emerge when both structure and parameter adaptation are required, with APM taking 5.5 ms and RPM taking 28.7 ms. In the final case, when two new variables are introduced, APM finds a revised model in 8.9 ms while RPM requires 32 ms. In all scenarios, revision is substantially faster than searching from scratch, with the greatest benefit occurring when only parameter revision is necessary.

Moreover, we expect this benefit will increase as more of the original model is retained. We can demonstrate this by adding variables to the base model while keeping constant the number of variables that differ in the target model. Figure 5 shows the CPU time to construct a model

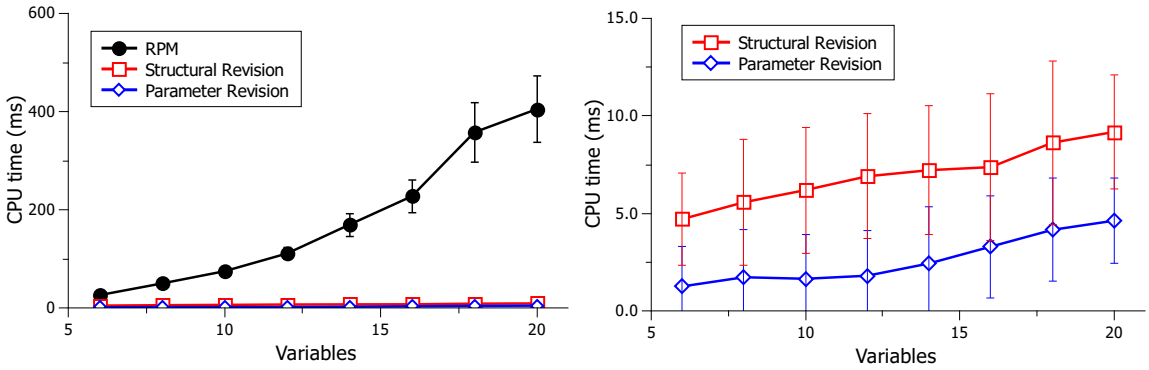

Figure 9. APM's processing time in milliseconds as a function of the number of variables in the target model when equations for variables require parameter revision and structure revision. The higher third curve in the left chart gives the time needed for RPM to induce the same models from scratch. The right chart shows a zoomed in view of the two revision curves.

from scratch using parametric revision and using structural revision. The first revision scenario in Subsection 4.1 corresponds to parametric revision and the second involves structural revision. In each case, the structure, parameters, and fits of the final models produced by RPM and APM are identical despite the differences in processing time.

Most important, the amount of benefit increases as APM retains more of the base model. RPM takes an average of 74 ms to construct a ten-variable model from scratch. Revising an existing model that has eight retained equations is roughly 45 times faster for parameter revision and 12 times for structure. For a 20-variable model the benefit is even greater. Revising an existing model with 18 variables retained is about 87 times faster when only parameters are estimated and some 44 times when structure revision is necessary. Both revision schemes show linear growth with the number of variables, while RPM appears to be polynomial in this factor.

We also find computational benefits when the system revises the more complex aquatic ecosystem model. The process library for this domain generates 43 process instances. RPM requires 555 ms to construct this five variable model from scratch by checking roughly 1,400 equations. In contrast, APM revises the parameters in one equation and the structure of another equation in 4 ms. In this case, only one equation required structural revision, so the system considered altering only those processes that influenced it, greatly reducing the number of candidate elements that can be included in the revised structure.

In summary, there are substantial computational savings when adapting an existing rate-based process model to explain new data, rather than inducing one from scratch. Parameter revision is the most efficient means, since it does not require search. When search for new equation structures is needed, the space is drastically reduced because it involves only a few variables and because the new equations must remain consistent with existing ones. As expected, the savings increase when more of the original model is retained, showing that revision of an existing model is far more scalable than constructing one from scratch.

5. Related Research on Model Revision

Our approach to adapting rate-based process models shares many ideas with previous efforts. For instance, the term *theory revision* has been used to describe a range of techniques that alter an existing model in response to new data, especially in the context of classifier learning. Mooney and Richards (1992) report one such method for automatically repairing a handcrafted logic program to cover supervised training cases. Towel, Shavlik, and Noordewier (1990) report another scheme that translates a logic program into a multilayer neural network, trains it on supervised data, and converts the result back into rules. These efforts focus on improving the classification accuracy of qualitative logical models, whereas APM instead addresses revision of explanatory scientific models for quantitative dynamic systems.

Another line of research, more closely related to our own, has applied similar ideas to revising explanatory scientific models. Darden (1990) proposes techniques for distinguishing different types of anomalies with respect to gene theory, which in turn have implications for revision. O'Rorke, Morris, and Schulenburg (1990) report a system that uses abduction to form hypotheses that it can revise if it encounters contradicting observations. Rajamoney (1990) describes an approach to explanation-based revision that changes an initial qualitative process model to assimilate anomalous observations. Each of these efforts involve the revision of explanatory models, but they focus on qualitative accounts rather than quantitative ones, as we have done.

A third paradigm for model adaptation draws on structural analogy. This technique uses existing knowledge about one situation to understand and make inferences about another. Falkenhainer (1990) reports a system that uses analogies with known process models to explain new scientific phenomena, although its adaptation abilities are limited. Friedman and Forbus (2010) expand on these ideas further, describing an explanation-based approach to conceptual change in process models that responds to new observations. These techniques revise qualitative models of scientific data, whereas our work has focused on adapting quantitative models.

In fact, several researchers have developed systems that revise quantitative models using scientific data. Both Todorovski et al. (2003) and Saito and Langley (2007) describe techniques for equation discovery that alter parameters and revise functional forms. In other work, Bay, Shrager, Pohorille, and Langley (2002) report a system that starts with a partial model of gene regulation stated as a linear causal model and then uses statistical regularities to alter its structure. These systems revise quantitative scientific models, but only for static scenarios, and they emphasize descriptive summaries of data rather than deeper explanations.

We use a simple threshold based on the r^2 values of the new and old data sets to detect anomalies. This is similar to drift detection methods described by Gama et al. (2004) that also uses thresholds to analyze streaming data, including the amount of data to consider. Bifet and Gavalda (2007) use adaptive windowing to automatically adjust the number of data points used to monitor a data stream in order to learn a new bayesian model, improving both detection sensitivity and computational efficiency. These methods involve detecting anomalous classifications while our system is concerned with identifying anomalies in regression equations.

Some prior work on inductive process modeling has addressed the adaptation task. Asgharbeygi, Langley, Bay, and Arrigo (2006) describe a system that begins with a quantitative process model that it revises in response to observed time series. Like other early research in this area, their

approach assumes a less constrained notion of process that does not partition rate expressions from derivatives, so it carries out a more extensive search through the space of model structures. Their system also relies on gradient descent search to estimate parameters. These make it both more computational expensive and less robust than the approach we have taken.

In summary, model adaptation has been studied in a number of guises. Some research has focused on logic programs for classification, while other efforts have dealt with qualitative process models. Most work on revising quantitative models has emphasized algebraic rather than differential equations. In contrast, the RPM system uses the formalism of rate-based process models to partition the adaptation task across derivative terms, which simplifies considerably anomaly detection, parameter reestimation, and structure alteration.

6. Concluding Remarks

In this paper, we presented a novel approach to the adaptation of rate-based process models and their implementation in the APM system. We reported this system in terms of three main activities: detection of anomalous derivatives, parameter reestimation, and structure modification. The first stage detects anomalies by comparing initial r^2 scores for each equation to r^2 scores calculated using new data. APM marks equations for revision if the ratio of r^2 scores falls below a threshold. Revision begins with parameter estimation, using multiple linear regression, followed by structural revision when this does not improve the fit sufficiently. Structural revision generates new process instances only for variables that require revision and starts with the equation that has the most processes already defined. These both reduce the size of the search space over that when inducing a model from scratch.

We demonstrated, using synthetic data from two domains, a six-variable linear food chain and a five-variable ecosystem with a feedback loop, that APM can successfully adapt a base model to explain new time series. Experiments also revealed the computational savings of adapting a model versus constructing one from scratch using RPM, an earlier system that also creates rate-based process models. These showed that the reduction in CPU time is greatest when only parameter estimation is needed, but there are even savings when structural adaptation must occur. Most of this benefit comes from a reduction in the search space, as attention is limited to process instances that include the anomalous derivatives. The times needed for both parameter and structure revision appear to grow linearly with the size of the base model; when starting from scratch, the time instead grows as a higher-order polynomial.

These initial results are encouraging, but we should extend the approach in a number of directions. One involves enhancing APM to operate on multivariate time series that arrive in a continuing stream. The system would monitor each variable for deviations from predicted values and invoke the revision process, possibly more than once, when it decides the situation has changed. Anomaly detection should be inexpensive and adaptation effort grows linearly with the number of equations changed. Rate-based process models make predictions directly about derivatives, so there is no need for simulation if one can estimate derivatives from observed values. However, we should take care that parameter revision is not overused, as frequently changing parameter values could compensate for structural changes.

We should also improve APM's method for revising the coefficients and structure of equations. The current system retains only the best-scoring equation for a given anomalous derivative, but this may not be consistent with the best candidate for a derivative examined later. Future versions should instead use a form of beam search to retain a set of best-scoring equations and then select a good combination once they have all been handled. We should also incorporate a similar strategy for deciding whether a revised equation is good enough; this should reduce the need for fine tuning of the r^2 ratio, which is currently somewhat sensitive. In addition, we should replace APM's exhaustive search through the space of equation structures with a heuristic scheme that selects rate terms to incorporate based on their contribution.

A final area for research involves process invention, which should prove useful when APM's methods for reestimating coefficients and altering equation structure are unable to explain new observations. In such cases, it could postulate entirely new processes, each of which has an algebraic rate expression and a set of proportional derivatives. However, recall that any variable which appears in the rate expression must also appear as a derivative and that, during revision, only anomalous derivatives can appear as influences. This should reduce the set of candidate processes substantially, making the discovery of new processes far more tractable in the context of revising models than when inducing them from scratch. Combined with the other extensions we have outlined, this should make APM a more flexible and powerful tool for adapting scientific models to new settings.

Acknowledgements

The research reported here was supported in part by Grant No. N00014-11-1-0107 from the US Office of Naval Research, which is not responsible for its contents. We thank Will Bridewell, Sašo Džeroski, Rich Morin, Son To, and Ljupčo Todorovski for discussions that led to the approach we have described.

References

Bay, S. D., Shrager, J., Pohorille, A., & Langley, P. (2002). Revising regulatory networks: From expression data to linear causal models. *Journal of Biomedical Informatics*, *35*, 289–297.

Bifet, A., & Gavalda, R. (2007). Learning from time-changing data with adaptive windowing. *Proceedings of the SIAM International Conference on Data Mining* (pp. 443–448). Philadelphia, PA: Society for Industrial and Applied Mathematics.

Borrett, S. R., Bridewell, W., Langley, P., & Arrigo, K. R. (2007). A method for representing and developing process models. *Ecological Complexity*, *4*, 1–12.

Bridewell, W., Langley, P., Todorovski, L., & Džeroski, S. (2008). Inductive process modeling. *Machine Learning*, *71*, 1–32.

Bridewell, W. & Langley, P. (2010). Two kinds of knowledge in scientific discovery. *Topics in Cognitive Science*, *2*, 36–52.

Crawford, J. M., Farquhar, A., & Kuipers, B. J. (1990). QPC: A compiler from physical models into qualitative differential equations. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 365–372). Boston, MA: AAAI Press.

Darden, L. (1990). Diagnosing and fixing faults in theories. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation*, 319–354. San Francisco: Morgan Kaufmann.

Džeroski, S., Langley, P., & Todorovski, L. (2007). Computational discovery of scientific knowledge. In S. Džeroski & L. Todorovski (Eds.), *Computational discovery of communicable scientific knowledge*. Berlin: Springer.

Džeroski, S., & Todorovski, L. (2008). Equation discovery for systems biology: Finding the structure and dynamics of biological networks from time course data. *Current Opinion in Biotechnology*, *19*, 360–368.

Falkenhainer, B. (1990). A unified approach to explanation and theory formation. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation*, 157–196. San Francisco: Morgan Kaufmann.

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

Friedman, S. E., & Forbus, K. D. (2010). An integrated systems approach to explanation-based conceptual change. *Proceedings of the Twenty-Fourth International Conference on Artificial Intelligence* (pp. 1523–1529). Atlanta: AAAI Press.

Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. In A. L. C. Bazzan & S. Labidi (Eds.), *Advances in artificial intelligence*, 286–295. Berlin: Springer.

Langley, P., Sanchez, J., Todorovski, L., & Džeroski, S. (2002). Inducing process models from continuous data. *Proceedings of the Nineteenth International Conference on Machine Learning* (pp. 347–354). Sydney: Morgan Kaufmann.

Langley, P., & Arvay, A. (2015). Heuristic induction of rate-based process models. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (pp. 537–543). Austin, TX: AAAI Press.

Mathews, J. H., & Fink, K. K. (2004). *Numerical methods using MATLAB* (4th ed.). Upper Saddle River, NJ: Prentice-Hall.

Mooney, R. J., & Richards, B. L. (1992). Automated debugging of logic programs via theory revision. *Proceedings of the Second International Workshop on Inductive Logic Programming* (pp. 1–16). Tokyo, Japan.

O'Rorke, P., Morris, S., & Schulenburg, D. (1990). Theory formation by abduction: A case study based on the chemical revolution. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation*, 197–224. San Francisco: Morgan Kaufmann.

Rajamoney, S. (1990). A unified approach to empirical discovery. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation*, 226–254. San Francisco: Morgan Kaufmann.

Shrager, J., & Langley, P. (Eds.) (1990). *Computational models of scientific discovery and theory formation*. San Francisco: Morgan Kaufmann.

Towell, G. G., Shavlik, J. W., & Noordewier, M. O. (1990). Refinement of approximate domain theories by knowledge-based neural networks. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 861–866). Boston: AAAI Press.