Integrated Systems for Inducing Spatio-Temporal Process Models

Chunki Park,[1] **Will Bridewell,**[1,2] and **Pat Langley**[1]

[1]Institute for the Study of Learning and Expertise
2164 Staunton Court, Palo Alto, CA 94306 USA
[2]Stanford Center for Biomedical Informatics Research
Stanford University, Stanford, CA 94305 USA

Abstract

Quantitative modeling plays a key role in the natural sciences, and systems that address the task of inductive process modeling can assist researchers in explaining their data. In the past, such systems have been limited to data sets that recorded change over time, but many interesting problems involve both spatial and temporal dynamics. To meet this challenge, we introduce SCISM, an integrated intelligent system which solves the task of inducing process models that account for spatial and temporal variation. We also integrate SCISM with a constraint learning method to reduce computation during induction. Applications to ecological modeling demonstrate that each system fares well on the task, but that the enhanced system does so much faster than the baseline version.

Introduction

Model building rests at the core of scientific inquiry, where it bridges the gap between theory and observation. Theoretical statements are too general to explain any particular system of interest, and observations only record that system's behavior. In contrast, a model consists of instantiated laws and auxiliary assumptions that characterize the important aspects of a complex system. When represented as a program, a model gives a formal, executable explanation of the studied phenomena. Moreover, one can use that program to explore potential system behavior, to analyze the effects of uncertainty, and to decide how to spend one's resources.

Inductive process modeling (Langley et al. 2002) is a scientific discovery task directed toward the automated modeling of complex, dynamic systems. Over the past decade, researchers have developed several approaches to address this problem (Asgharbeygi et al. 2006; Todorovski et al. 2005; Bridewell et al. 2008). Each has taken the form of an integrated system that combines search for model structures and a method for parameter estimation, along with a performance element that simulates individual models. These systems have successfully addressed problems in ecology (Asgharbeygi et al. 2006), biology (Langley et al. 2006), and other disciplines.

Although researchers have made substantial progress on this task, their methods to date have been limited to modeling change over time and have not simultaneously accounted for spatial variation. As a result, their systems cannot address many problems that interest scientists working in climatology, systems biology, population ecology, and immunology. In response to this limitation, we introduce the task of learning scientific process models of spatio-temporal dynamics. We also report two integrated systems that handle this task and demonstrate their effectiveness on an ecological modeling problem.

These two systems build on the same general architecture of earlier approaches. The first, SCISM,[1] adapts previous methods for structural and parametric search to handle spatial model induction. The second, which we call SCISM′, extends it by introducing a third level of search. This layer dynamically alters the structural search space to rule out large classes of models unlikely to explain a given data set. Experiments show incorporating this technique leads to considerable reduction in computational costs.

In the following section, we review inductive process modeling in more detail, pose the new problem of learning spatio-temporal process models, and introduce SCISM as an initial approach to this task that handles one spatial dimension. Next we describe the new learning component and its integration with SCISM to produce SCISM′. We then report results for both systems, discuss related and future work, and offer some closing thoughts.

An Integrated Process Modeler

Scientists and engineers often state their models in terms of *processes* that govern system dynamics and *entities* that are changed by those processes. Entities collect related properties, such as variables and constants, that describe the current state, whereas processes relate entities and determine how their properties change. A *process model* is a collection of interlinked processes and entities that explain the behavior of a dynamical system. For example, a process model of population dynamics might include predators and prey, such as foxes and rabbits. These species are the entities and their properties include population density and growth rates. Processes for growth, death, and predation connect the rabbits to the foxes and explain how the species interact.

[1]**S**tructurally **C**onstrained **I**nduction of **S**patial **M**odels

Model builders rarely start from scratch. Instead, they rely on generic structures that are useful across situations. We refer to this background knowledge as a collection of *generic processes* and *generic entities*. These vary across domains and act as templates for the instantiated components of a particular process model. The generic processes include entity roles that are filled by specific entities from the modeling problem and parameters that let them adapt to different scenarios. The generic entities indicate the properties that play a part in system dynamics.

The task of *inductive process modeling* (Bridewell et al. 2008) uses this domain knowledge to explain the behavior of a dynamical system. More specifically, we can state the task as:

- *Given*: Generic entities that have properties relevant to the observed dynamics;
- *Given*: Generic processes that specify causal relations among entities using generalized functional forms;
- *Given*: A set of entities present in the modeled system;
- *Given*: Observations for the continuous properties of those entities as they change over time;
- *Find*: A process model that explains the observed data and predicts unseen data accurately.

Previous efforts have produced several systems for inductive process modeling that search for quantitative models to explain temporal dynamics (Asgharbeygi et al. 2006; Bridewell et al. 2008; Langley et al. 2002). Recent approaches have added structural constraints to the task description as a way to reduce the number of models considered (Todorovski et al. 2005). In this paper, we extend this task so that it requires explanation of observations for continuous variables as they vary over both time and space. As we describe later, this requires changes to the model representation, performance element, and learning mechanism.

As the starting point for this research, we use SC-IPM (Bridewell and Langley 2010), an integrated system that uses modular constraints to guide process model induction. In the remainder of this section, we describe SCISM, which extends SC-IPM to handle the extended task of inducing process models that account for both temporal variation and spatial variation in one dimension.

Quantitative Process Models

As in previous work, SCISM learns *quantitative process models*, which encode qualitative structure as a set of processes that contain quantitative forms of algebraic and ordinary differential equations (ODEs). However, modeling problems often involve a spatial component that affects their dynamics (e.g., Arrigo et al. 2008; Zador et al. 1994) and that rely on partial differential equations (PDEs) to capture this aspect. Unlike ODEs, which are functions of a single independent variable (e.g., time), PDEs can be functions of multiple variables and can include partial derivatives taken with respect to them. This feature lets PDE-based models explain system dynamics in terms both of time and of spatial variables, such as depth or position on an x–y coordinate plane. To handle this richer collection of modeling problems, we extended quantitative process models to support partial differential equations.

Table 1 depicts an example quantitative process model for an aquatic ecosystem and shows how we extended the formalism to include PDEs. The model refers to four entities: phytoplankton (phy), zooplankton (zoo), nitrate (nit), and detritus (det). These entities appear in the first four processes that define loss due to death of phytoplankton and zooplankton, the grazing interaction between those two species, and the uptake of nitrates by phytoplankton. The last two processes, which include partial derivatives, characterize vertical movement of plankton in the water column.

The processes *phy_spatial_dynamics* and *zoo_spatial_dynamics* in Table 1 illustrate our extension to the process modeling language. To represent spatial dynamics, we let a process include partial derivatives in its equations as in $d[phy.conc, t, 1] = -0.65 * d[phy.conc, z, 1]$, which specifies the concentration of phytoplankton. The independent variables are time, t, and depth, z, with their partial derivatives represented as $d[phy.conc, t, 1]$ for $\partial(phy.conc)/\partial t$ and $d[phy.conc, z, 1]$ for $\partial(phy.conc)/\partial z$. The number '1' inside brackets indicates a first-order derivative.

The performance task involves simulating a model to produce a trajectory that one can compare to observations. One component of SCISM carries out this simulation, first transforming the process model into a system of algebraic and differential equations, and then using numerical methods to solve the equations. Generating equations involves combining the effects from each process. By default, SCISM adds these effects, but it can also take their product, maximum, or minimum according to variable-specific information stored in the generic entities. Eq. (1) lists the system of equations built by summing the effects of each process.

$$\begin{aligned}
\frac{\partial P}{\partial t} &= -0.65 * \frac{\partial P}{\partial z} + (0.5 - 0.05) * P - 0.15 * Z \; , \\
\frac{\partial Z}{\partial t} &= -0.75 * \frac{\partial Z}{\partial z} + (0.3 * 0.15 - 0.1) * Z \; , \\
\frac{\partial N}{\partial t} &= -0.01 * 0.5 * P \; , \\
\frac{\partial D}{\partial t} &= 0.05 * P + (0.1 + (1 - 0.3) * 0.15) * Z \; ,
\end{aligned} \tag{1}$$

where P, Z, N, and D signify $phy.conc$, $zoo.conc$, $nit.conc$, and $det.conc$, respectively.

To solve systems of PDEs, SCISM relies on the semi-discrete method (Heath 2001), which discretizes the spatial dimensions but leaves time continuous. Using this method, the simulator breaks up each PDE into a set of ODEs whose solutions approximate those of the original equation. Starting with initial conditions from a data set, SCISM's simulator solves the ODEs for the second time point. Output from this step serves as input to solve equations for the third time point. This procedure iterates until it produces a complete trajectory that one can analyze or compare to observations.

Inducing Quantitative Process Models

Building explanatory models requires background knowledge about the world and phenomena to be explained. In

Table 1: A process model for an aquatic ecosystem.

```
model Aquatic_Ecosystem
  entities : $phy, zoo, nit, det$
  process phy_loss
    equations : $d[phy.conc, t, 1] = -1 * 0.05 * phy.conc$
                $d[det.conc, t, 1] = 0.05 * phy.conc$
  process zoo_loss
    equations : $d[zoo.conc, t, 1] = -1 * 0.1 * zoo.conc$
                $d[det.conc, t, 1] = 0.1 * zoo.conc$
  process zoo_phy_grazing
    equations : $d[zoo.conc, t, 1] = 0.3 * 0.15 * zoo.conc$
                $d[det.conc, t, 1] = (1 - 0.3) * 0.15 * zoo.conc$
                $d[phy.conc, t, 1] = -1 * 0.15 * zoo.conc$
  process nit_uptake
    equations : $d[phy.conc, t, 1] = 0.5 * phy.conc$
                $d[nit.conc, t, 1] = -1 * 0.01 * 0.5 * phy.conc$
  process phy_spatial_dynamics
    equations : $d[phy.conc, t, 1] = -0.65 * d[phy.conc, z, 1]$
  process zoo_spatial_dynamics
    equations : $d[zoo.conc, t, 1] = -0.75 * d[zoo.conc, z, 1]$
```

Table 2: Some generic processes for an aquatic ecosystem.

```
generic process exponential_loss
  entities : S{species}, D{detritus}
  parameters : $\alpha[0, 1]$
  equations : $d[S.conc, t, 1] = -1 * \alpha * S.conc$
              $d[D.conc, t, 1] = \alpha * S.conc$
generic process grazing
  entities : S1{species}, S2{species}, D{detritus}
  parameters : $\rho[0, 1], \gamma[0, 1]$
  equations : $d[S1.conc, t, 1] = \gamma * \rho * S1.conc$
              $d[D.conc, t, 1] = (1 - \gamma) * \rho * S1.conc$
              $d[S2.conc, t, 1] = -1 * \rho * S1.conc$
generic process nutrient_uptake
  entities : S{species}, N{nutrient}
  parameters : $\beta[0, 1], \mu[0, 1]$
  equations : $d[S.conc, t, 1] = \mu * S.conc$
              $d[N.conc, t, 1] = -1 * \beta * \mu * S.conc$
generic process spatial_dynamics
  entities : S{species}
  parameters : $\kappa[0, 5]$
  equations : $d[S.conc, t, 1] = -\kappa * d[S.conc, z, 1]$
```

inductive process modeling, the phenomena are characterized by spatio-temporal data and the knowledge is encoded in a library of generic processes and entities. As we mentioned earlier, a generic process differs from a *specific* process in that it does not commit to specific entities or parameter values. Table 2 lists a partial process library for aquatic ecosystems. Notice that the process *exponential_loss* has two entity roles, S and D, and one parameter, α, whose value lies in the [0,1] interval. The specific processes *phy_loss* and *zoo_loss* in Table 1 instantiates *exponential_loss* with the entities for phytoplankton and zooplankton.

Once provided with background knowledge, which can also include structural constraints (Bridewell and Langley 2010) and a data set, SCISM calls its learning component to search through the space of possible models. This part of the system integrates an algorithm for exploring the space of model structures with one for estimating the parameters of a particular structure. The combined procedure for model generation is broken into three parts that:

1. generate all possible instantiations of generic processes with specific entities but without parameter values;
2. combine instantiated processes to form a generic model that satisfies all the structural constraints; and
3. estimate the parameter values and scores each model's fit to the data.

After carrying out this search, the system returns the quantitative process model that best accounts for the data.

Initial results with SCISM suggest that it can learn accurate models that include spatial components, but execution is much slower than systems like SC-IPM that are limited to ODEs. The primary cause is that the PDE solver, which is called by the parameter estimation component, must calculate multiple spatial values for each time point. There are four ways to reduce the total execution time for SCISM. The first two involve efficiency improvements to the numerical methods for parameter estimation and simulation. The third is to learn constraints that reduce the size of the search space, and the fourth is to transform the modeling problem into one suitable for ODEs, using the results to limit search for PDEs.

Let us consider each of these approaches in turn. Although improvements to the first solution would have far-reaching effects, SCISM already incorporates a state-of-the-art conjugate-gradient method (Bunch, Gay, and Welsch 1993) and progress on parameter estimation for nonlinear dynamical systems has been slow. We could improve the efficiency of the simulation routine, but the most effective techniques rely on problem-specific features of the equations and we would like to keep SCISM as general as possible. This leaves the third and fourth approaches. We incorporate both into SCISM, supporting them with a new component that learns structural constraints from previous modeling experience (Bridewell and Todorovski 2007).

Extending the SCISM System

By extending SCISM to dynamically reduce its search space during execution, we expect to substantially reduce overall computation. The strategy that we will use involves fitting ODE models to slices of the spatial data and learning constraints that rule out inaccurate model structures. In this section we describe the constraint induction system and describe how we integrate that component with SCISM.

The MISC Module

In previous work, Bridewell and Todorovski (2007) have described MISC,[2] a system that induces constraints on process models. To this end, it inputs logical descriptions of models in the search space, the scores of those models on a particular data set, and a logical description of the generic

[2]**M**ethod for **I**nducing **S**tructural **C**onstraints

Table 3: A partial list of the relational features that may appear in a constraint. Each predicate also has an explicitly defined negation.

includes_process(model_id, generic_process)
includes_process_group(model_id, process_group)
includes_entity_instance(model_id, entity_instance)
includes_entity(model_id, generic_entity)

process library. To generate the first two inputs, one runs an inductive process modeler and stores all model structures produced during model search, along with measures of accuracy in terms of coefficients of determination (r^2). MISC then converts the stored information into basic logical descriptions that identify which processes occur in each model, which entities appear in those processes, and each model's associated r^2 value.

MISC interprets the model descriptions using the domain-specific generic process library and a domain-independent set of relational features. The domain-specific information consists of the names of the generic processes, the types of entities that they relate, and groups of processes created from the constraints. For example, MISC would transform a mutual-exclusion constraint on growth processes into a group that contained exponential, logistic, and limited growth. Complementarily, relational features like those shown in Table 3 connect the specific processes and entities with their generic counterparts to describe the high-level structure of a model.

To learn constraints, MISC sets a threshold for labeling models as accurate or inaccurate, then uses Aleph (Srinivasan 2004), an inductive logic programming system, to carry out supervised learning. One can select the threshold using several different methods, but we describe a specific technique in the next section. Aleph generates rules that classify models as either inaccurate or accurate. Two such learned constraints are:

```
accurate_model(M) :-
    does_not_include_process_group(M, limited_growth),
    includes_process(M, holling_type_3).

inaccurate_model(M) :-
    includes_process(M, nut_lim_exp),
    includes_process(M, ratio_dependent_2).
```

The first rule, stated in Prolog notation, says that accurate models lack a process from the $limited\_growth$ group and include the $holling\_type\_3$ predation process. The second rule says that models which include instances of the generic processes nut_lim_exp and $ratio\_dependent\_2$ are inaccurate. One can automatically convert these rules into constraints for use by inductive process modeler.

Prior work shows that constraints learned from a system like MISC can reduce the number of structures considered during search by an order of magnitude (Bridewell and Todorovski 2007). However, the same work also suggests that the rules for accurate models can be overly restrictive, so here we use only those constraints based on the rules for inaccurate models. We designed this approach as a stand-alone method for transfer learning, but we realized that integrating it with SCISM could make spatio-temporal modeling more tractable.

Integrating MISC with SCISM

To reduce search through the space of spatio-temporal models, we integrated MISC with SCISM. For convenience, we refer to this new system as SCISM$'$. The intuition behind this extension is that, when the spatial dynamics of a PDE are numerically stable (i.e., not stiff), the ODE formed by removing partial derivatives provides a reasonable estimate of the solution. Therefore, we should be able to solve several regular inductive process modeling problems, apply MISC to learn constraints, and then search the reduced space of PDE models.

To illustrate this idea, consider the PDE from Eq. 1,

$$\frac{\partial P}{\partial t} = -0.65\frac{\partial P}{\partial z} + (0.5 - 0.05)P - 0.15Z \ . \qquad (2)$$

One can discretize the single spatial dimension, z, by taking spatial slices of the training data, while still treating time, t, as continuous. In the first slice of the data set where $z = z_0$, we create the ODE $\frac{dP}{dt}|_{z=z_0} \approx (0.5 - 0.05)P - 0.15Z$ by disregarding the partial derivative $\frac{\partial P}{\partial z}$. Repeating this procedure for all spatial slices (e.g., for each depth recorded in the data set) creates a set of ODEs whose solution approximates that of the original PDE. By learning structural constraints after modeling each slice, SCISM$'$ can systematically reduce the number of candidate solutions in each subsequent search.

Integrating this added layer of search into SCISM$'$ requires slight changes to the input and substantial changes to SCISM's implementation. First, the input differs in that SCISM$'$ needs to know how many spatial slices it should use. Oceanographic data may have measurements from several depths, but modeling the dynamics at only a few may constrain the PDE search sufficiently. The system also needs a function for calculating the threshold used by MISC. Recall that we utilize this number to label candidate models as either accurate or inaccurate for use during constraint induction. Currently, we assume that models with an r^2 greater than 0.7 are accurate and those below 0.7 are inaccurate.

During its initial cycle, SCISM$'$ searches for ODE-based process models that fit the first spatial slice, storing all the candidates that it evaluates. MISC takes these candidates as input and learns a set of rules that classify the inaccurate models. A separate component transforms these rules into constraints for process modeling. For example, if MISC finds the rule

inaccurate model(M) :-
  includes_process$(M, process_1)$,
  includes_process_group$(M, group_1)$,
  does_not_include_process_group$(M, group_2)$,

then SCISM$'$ produces the three Boolean constraints
  $\neg process\_1$,
  $\neg process\_2 \wedge \neg process\_3$,
  $process\_4 \vee process\_5 \vee process\_6$,

based on the knowledge that $group\_1$ includes $process\_2$

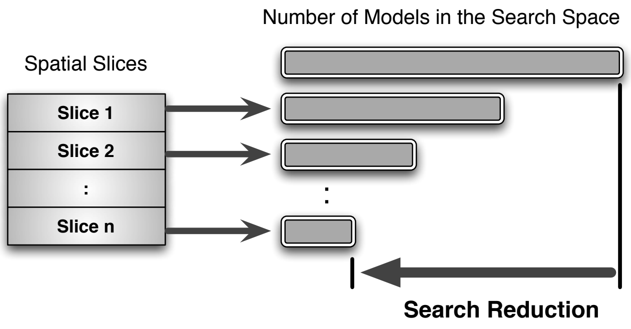

Figure 1: In SCISM′, the number of models in the search space decreases as constraints accumulate. In SCISM the size of the search space is identical to the top bar, but the required computation is equal to full bars for each slice.

and $process\_3$, while $group\_2$ includes $process\_4$, $process\_5$, and $process\_6$. The system then uses the constraints to limit search when modeling the next spatial slice. This procedure iterates through all the slices, ending with a search through the space of quantitative process models that include partial derivatives. Figure 1 illustrates how this approach reduces the number of structures considered at each layer. Figure 2 summarizes the SCISM′ framework, including communication channels between SCISM and MISC.

Inducing Spatio-Temporal Models

The previous sections introduced SCISM and SCISM′, two systems that solve the expanded task of inducing process models with both temporal and spatial dynamics. In this section, we demonstrate their effectiveness and report on the difference in their computation costs.

To study the ability of SCISM and SCISM′ to build spatio-temporal models, we started with a hand-constructed model of an aquatic ecosystem that incorporates one-dimensional spatial dynamics within a water column in addition to changes over time. The model has three entities for phytoplankton, zooplankton, and nitrate, and it includes ten processes that control growth, grazing, nutrient mixing, and other interactions. We simulated the model to create a synthetic data set with 75 measurements for three variables at ten depth levels. To perturb the data, we added noise to each trajectory by replacing each generated value v with $(1 + 0.1g)v$, where g was a Gaussian random variable with a mean of 0 and a standard deviation of 1.

First we determined that SCISM and SCISM′ can recover the original model from these data. To this end, we took the complete trajectories and split them into three smaller sets, containing 50 time points each, to mimic more realistic data sets with missing values. We then ran both systems on each set of trajectories with a library of 24 generic processes originally developed with the assistance of Earth scientists. This library generated a total of 672 candidate model structures. In every case, the highest scoring model found by SCISM and SCISM′ had a structure identical to the original model. There was some variation in the parameter values, but we expected this given the perturbations that we introduced. The average mean–squared-error for SCISM was 0.25 with a standard error of 0.08. The average for SCISM′ was 0.21 with a standard error of 0.06. The slightly better scores for SCISM′ are attributable to variation in the parameter estimation routine.

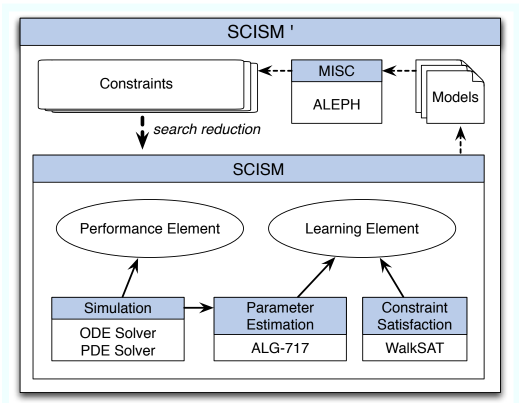

Figure 2: The frameworks of SCISM and SCISM′.

Next, we compared the time taken by each program to determine how much SCISM′ reduced the search space for particular trials. We ran all experiments on machines with Intel Core 2 Duo processors operating at 3.15GHz, with the exception that we ran the MISC component on a 2.33GHz Core 2 Duo processor.[3] SCISM finished in an average of 15.35 hours of processor time and SCISM′ took on average 4.73 hours, with 3.33 hours for the SCISM component and 1.40 hours for MISC. Thus, there was more than a three-fold reduction of computation after integrating MISC into the process modeling system.

When running SCISM′ we selected three depths to fit with ODE-based models. After running on the first slice, the system found eight constraints and reduced the space of candidates from 672 structures to 144, a reduction of 78%. After the second slice, SCISM′ found one new constraint, which further reduced the number of candidates to 128, but it learned no new constraints from the third slice. The system then used SCISM to search through the reduced space of spatio-temporal process models.

These experiments show not only that one can build systems to solve the task of inducing spatio-temporal process models, but that integrating the ability to transfer learned constraints across spatial slices substantially improves efficiency over the baseline approach. Overall, SCISM′ reduced expensive search through PDE-based models by more than 80% over SCISM and it cut execution time by nearly 70%.

Discussion

Our research on SCISM and SCISM′ draws on but extends previous work on equation discovery. For instance, Bradley, Easley, and Stolle's (2001) PRET used constraints in constructing differential equation models from time series, but it relied on qualitative analysis to limit search and it could induce only a single equation. Moreover, it could not handle spatio-temporal data. Another equation discovery system,

[3]We separated resources in this manner due to software availability on the compute cluster.

Todorovski and Džeroski's (1997) LAGRAMGE, is closely related to inductive process modelers. Like SC-IPM and the systems introduced here, it found models composed of multiple equations that govern a dynamic systems' behavior. However, it could not build models with hidden variables and it did not construct process explanations.

SCISM′ has even more in common with Todorovski's (2003) PADLES, which discovered models stated as sets of partial differential equations. Like SCISM′, this system discretizes spatial components and models each slice with ordinary differential equations. After this step, PADLES stores the n highest scoring models for each slice, selects the frequently occurring ones, adds the needed spatial derivatives, and estimates the parameters for this reduced list of model structures. In contrast, SCISM′ uses a more sophisticated method for reducing search, accumulating constraints based on modeling experience rather than selecting a fixed number of models to explore.

SCISM′ constructs promising models in the reduced search space, but there is room for improvement. For instance, the system currently uses rules about inaccurate models to develop its constraints because rules for accurate models tend to be overly restrictive and are, unintuitively, more likely to rule out the most promising candidates. Finding a way to use the constraints effectively could lead to further reductions in search. Additionally, we need to extend the system to work in more than one spatial dimension. Here the primary concern is computational efficiency when solving the underlying equations, since the system's semi-discrete methods scale exponentially with the number of spatial dimensions. There are two standard approaches to this problem: (1) handle the growth through parallel computation or (2) control it by increasing the coarseness of the simulation. Although increased coarseness can reduce accuracy, the results may guide the system in deciding when to give a model structure a closer look. As a third option, we could recursively apply the method used in SCISM′ to learn additional constraints as the system adds spatial dimensions.

In this paper, we introduced the problem of inducing spatio-temporal process models and described SCISM, a system that addresses this task in the presence of one spatial dimension. We also demonstrated that integrating SCISM with MISC, a system for learning constraints on process models, leads to substantial reduction in computational costs while retaining the ability to recover the correct model from noisy data. These results indicate that SCISM′ is not only an interesting example of integrated intelligence but also offers a promising approach to aiding scientists in explaining spatial-temporal observations.

Acknowledgments

The research reported in this paper was supported by NSF Grant Number IIS-0326059 and occurred while the first author was a graduate student at Stanford University. We thank Kevin Arrigo and Stuart Borrett for their advice on the process library for aquatic ecosystems and Ljupčo Todorovski for his contributions to MISC.

References

Arrigo, K. R.; van Dijken, G. L.; and Bushinsky, S. 2008. Primary production in the southern ocean, 1997–2006. *Journal of Geophysical Research* 113:C08004.

Asgharbeygi, N.; Langley, P.; Bay, S.; and Arrigo, K. 2006. Inductive revision of quantitative process models. *Ecological Modelling* 194:70–79.

Bradley, E.; Easley, M.; and Stolle, R. 2001. Reasoning about nonlinear system identification. *Artificial Intelligence* 133:139 – 188.

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

Bridewell, W., and Todorovski, L. 2007. Learning declarative bias. In *Proceedings of the Seventeenth Annual International Conference on Inductive Logic Programming*, 63–77. Corvallis, OR: Springer.

Bridewell, W.; Langley, P.; Todorovski, L.; and Džeroski, S. 2008. Inductive process modeling. *Machine Learning* 71:1–32.

Bunch, D. S.; Gay, D. M.; and Welsch, R. E. 1993. Algorithm 717: Subroutines for maximum likelihood and quasi-likelihood estimation of parameters in nonlinear regression models. *ACM Transactions on Mathematical Software* 19:109–130.

Heath, M. T. 2001. *Scientific Computing*. McGraw-Hill Higher Education.

Langley, P.; Sánchez, J.; Todorovski, L.; and Džeroski, S. 2002. Inducing process models from continuous data. In *Proceedings of the Nineteenth International Conference on Machine Learning*, 347–354. San Francisco, CA: Morgan Kaufmann.

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

Srinivasan, A. 2004. *The Aleph Manual*. Technical Report, Computing Laboratory, Oxford University, Oxford, United Kingdom.

Todorovski, L., and Džeroski, S. 1997. Declarative bias in equation discovery. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 376–384. San Francisco: Morgan Kaufmann.

Todorovski, L.; Bridewell, W.; Shiran, O.; and Langley, P. 2005. Inducing hierarchical process models in dynamic domains. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 892–897. Pittsburgh: AAAI Press.

Todorovski, L. 2003. *Using domain knowledge for automated modeling of dynamic systems with equation discovery*. Ph.D. Dissertation, University of Ljubljana, Ljubljana, Slovenia.

Zador, A., and Koch, C. 1994. Linearized models of calcium dynamics: Formal equivalence to the cable equation. *Journal of Neuroscience* 14:4705–4715.