Computational Discovery of Scientific Models: Guiding Search with Knowledge and Data

Pat Langley

Institute for the Study of
Learning and Expertise

Department of Computer Science
University of Auckland

Thanks to K. Arrigo, A. Arvay, G. Bradshaw, S. Borrett, W. Bridewell, S. Dzeroski, H. Simon, L. Todorovski, and J. Zytkow for their contributions to this research, which was funded by NSF Grant No. IIS-0326059 and ONR Grant No. N00014-11-1-0107.

Background

Science and Discovery

Science is a diverse collection of activities that is distinguished by important characteristics:

- Systematic collection and analysis of observations
- Formal statement of theories, laws, and models
- Use of theories / models to explain and predict observations
- Use of observations to evaluate theories / models

Scientific discovery is widely viewed as one of the highest forms of human achievement.

Computational accounts of such discovery would have important implications, both theoretical and practical.

Mystical Views of Discovery

Most philosophers have avoided scientific discovery, believing it immune to logical analysis. Popper (1934) wrote:

> *The initial stage, the act of conceiving or inventing a theory, seems to me neither to call for logical analysis nor to be susceptible of it … My view may be expressed by saying that every discovery contains an 'irrational element', or 'a creative intuition' …*

Hempel and many others also believed discovery was inherently irrational and beyond understanding.

However, advances made by two fields – *cognitive psychology* and *artificial intelligence* – in the 1950s suggested otherwise.

Scientific Discovery as Problem Solving

Simon (1966) offered another view – scientific discovery is a variety of *problem solving* that involves:

- *Search* through a space of *problem states*
- Generated by applying mental *operators*
- Guided by *heuristics* to make it tractable

Heuristic search had been implicated in many cases of human cognition, from proving theorems to playing chess.

This framework offered not only a path to understand scientific discovery, but also ways to *automate* this mysterious process.

Early Progress

For my CMU dissertation research, I adapted Simon's ideas on scientific discovery, developing a computer program that:

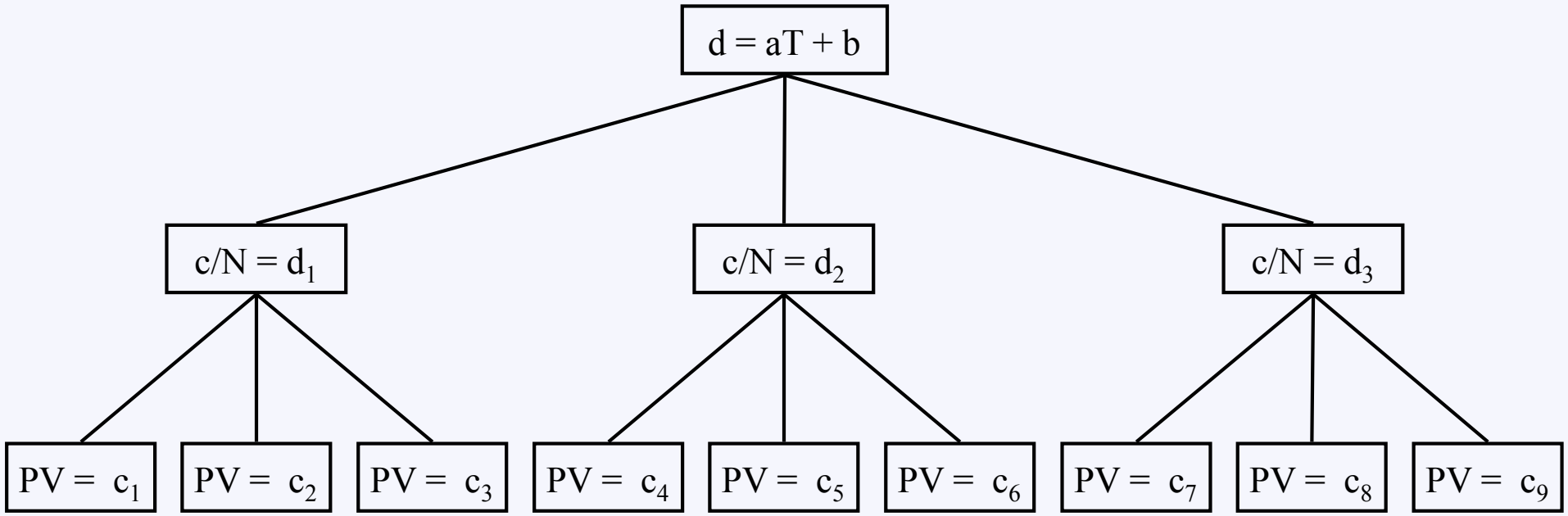
- Carried out search in a problem space of theoretical terms;
- Using operators that combined old terms into new ones;
- Guided by heuristics that noted regularities in data; and
- Applied these recursively to formulate higher-level relations.

The result was *Bacon* (Langley, 1981), an early AI system that rediscovered laws from the history of physics and chemistry.

I named the system after Sir Francis Bacon because it adopted a data-driven approach to discovery.

Bacon on the Ideal Gas Law

Bacon rediscovered the ideal gas law, PV = aNT + bN, in three stages, each at a different level of description.



Parameters for laws at one level became dependent variables in laws at the next level, enabling discovery of complex relations.

Some Laws Discovered by Bacon

Basic algebraic relations:

- Ideal gas law $PV = aNT + bN$
- Kepler's third law $D^3 = [(A - k) / t]^2 = j$
- Coulomb's law $FD^2 / Q_1Q_2 = c$
- Ohm's law $TD^2 / (LI - rI) = r$

Relations with *intrinsic properties*:

- Snell's law of refraction $\sin I / \sin R = n_1 / n_2$
- Archimedes' law $C = V + i$
- Momentum conservation $m_1V_1 = m_2V_2$
- Black's specific heat law $c_1m_1T_1 + c_2m_2T_2 = (c_1m_1 + c_2m_2)T_f$

Ensuing Systems for Law Discovery

Bacon inspired other many AI systems for law discovery like:

- ABACUS (Falkenhainer, 1985) and ARC (Moulet, 1992)
- Fahrenheit (Zytkow, Zhu, & Hussam, 1990)
- COPER (Kokar, 1986) and E* (Schaffer, 1990)
- IDS (Nordhausen & Langley, 1990)
- Hume (Gordon & Sleeman, 1992)
- DST (Murata, Mizutani, & Shimura, 1994)
- SSF (Washio et al., 1997) and LaGramge (Todorovski et al., 2006)
- GP (Koza et al., 2001) and Eureqa (Schmidt & Lipson, 2009)

These relied on different methods but also searched for explicit mathematical laws that matched data.

Successes of Computational Scientific Discovery

AI systems of this type have helped to discover new knowledge in many scientific fields:

- reaction pathways in catalytic chemistry (Valdes-Perez, 1994, 1997)
- qualitative chemical factors in mutagenesis (King et al., 1996)
- quantitative laws of metallic behavior (Sleeman et al., 1997)
- quantitative conjectures in graph theory (Fajtlowicz et al., 1988)
- qualitative conjectures in number theory (Colton et al., 2000)
- temporal laws of ecological behavior (Todorovski et al., 2000)
- models of gene-influenced metabolism in yeast (King et al., 2009)

Each of these has led to publications in the *refereed literature of the relevant scientific field*.

The Data Mining Movement

During the 1990s, a new research paradigm – known as *data mining* – emerged that:

- Emphasized the availability of large amounts of data
- Used computational methods to find regularities in the data
- Adopted heuristic search through a space of hypotheses
- Initially focused on commercial applications and data sets

Most research adopted notations invented by computer scientists, unlike scientific discovery, which used *scientific formalisms*.

Data mining has been applied to scientific data, but the results seldom bear a resemblance to scientific *knowledge*.

Discovering Explanatory Models

Discovering Explanatory Models

The early stages of any science focus on *descriptive laws* that *summarize* empirical regularities.

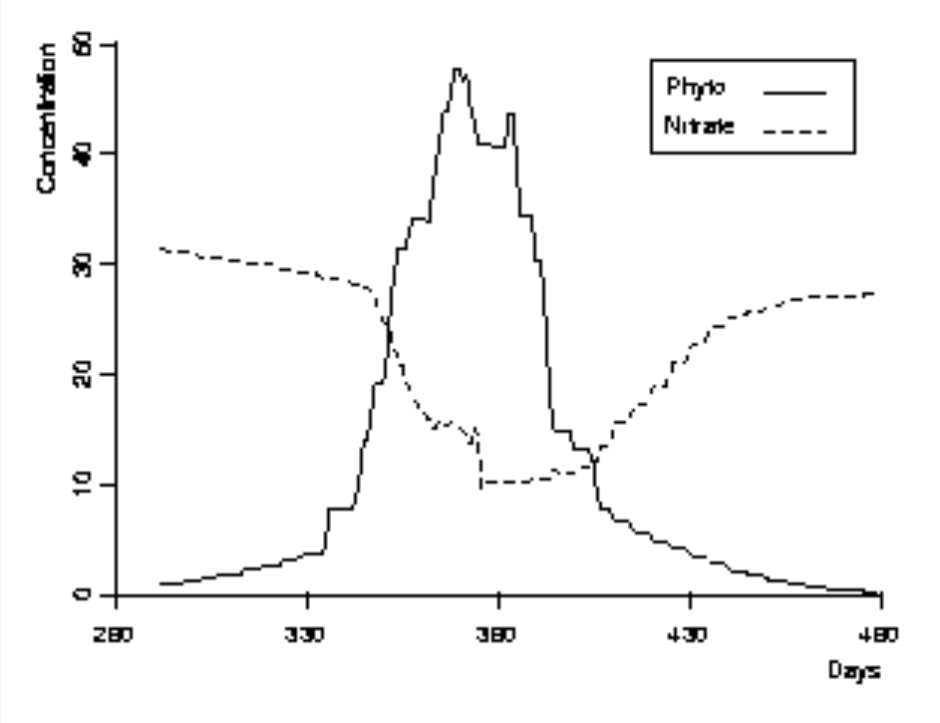
Mature sciences instead emphasize the creation of *models* that *explain* phenomena in terms of:

- Inferred *components* and *structures* of entities
- Hypothesized *processes* about entities' interactions

Explanatory models move beyond description to provide deeper accounts linked to theoretical constructs.

Can we develop computational systems that address this more sophisticated side of scientific discovery?

An Example: The Ross Sea Ecosystem



Formal accounts of ecosystem dynamics are often cast as sets of differential equations.

Here four equations describe the concentrations of phytoplankton, zooplankton, nitrogen, and detritus in the Ross Sea over time.

Such models can match observed variables with some accuracy.

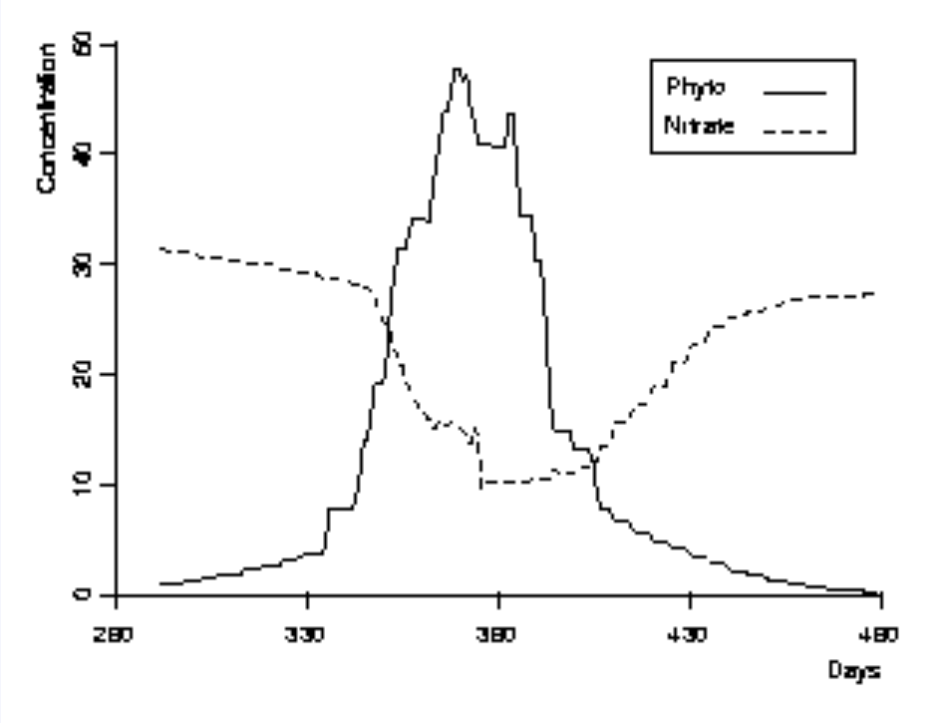
$$d[phyto,t,1] = -0.307 \times phyto - 0.495 \times zoo + 0.411 \times phyto$$

$$d[zoo,t,1] = -0.251 \times zoo + 0.615 \times 0.495 \times zoo$$

$$d[detritus,t,1] = 0.307 \times phyto + 0.251 \times zoo + 0.385 \times 0.495 \times zoo - 0.005 \times detritus$$

$$d[nitro,t,1] = -0.098 \times 0.411 \times phyto + 0.005 \times detritus$$

A Deeper Account of Ross Sea Dynamics



As phytoplankton uptakes nitrogen, its concentration increases and the nitrogen decreases. This continues until the nitrogen is exhausted, which leads to a phytoplankton die off. This produces detritus, which gradually remineralizes to replenish nitrogen. Zooplankton grazes on phytoplankton, which slows the latter's increase and also produces detritus.

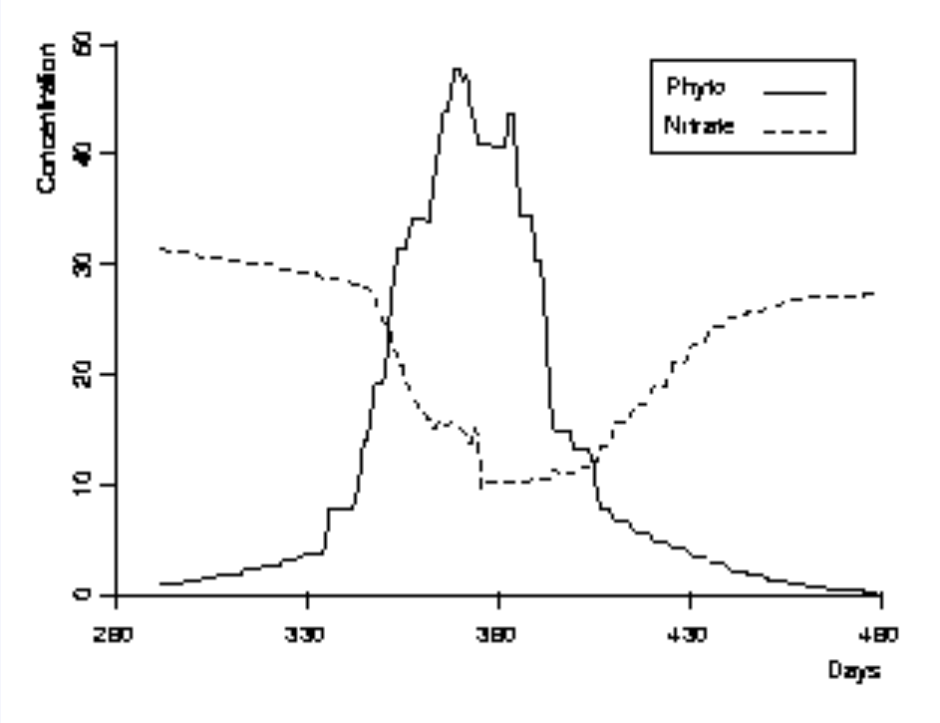
$$d[phyto,t,1] = -0.307 \times phyto - 0.495 \times zoo + 0.411 \times phyto$$

$$d[zoo,t,1] = -0.251 \times zoo + 0.615 \times 0.495 \times zoo$$

$$d[detritus,t,1] = 0.307 \times phyto + 0.251 \times zoo + 0.385 \times 0.495 \times zoo - 0.005 \times detritus$$

$$d[nitro,t,1] = -0.098 \times 0.411 \times phyto + 0.005 \times detritus$$

Processes in Ross Sea Dynamics



As phytoplankton uptakes nitrogen, its concentration increases and the nitrogen decreases. This continues until the nitrogen is exhausted, which leads to a phytoplankton die off. This produces detritus, which gradually remineralizes to replenish nitrogen. Zooplankton grazes on phytoplankton, which slows the latter's increase and also produces detritus.

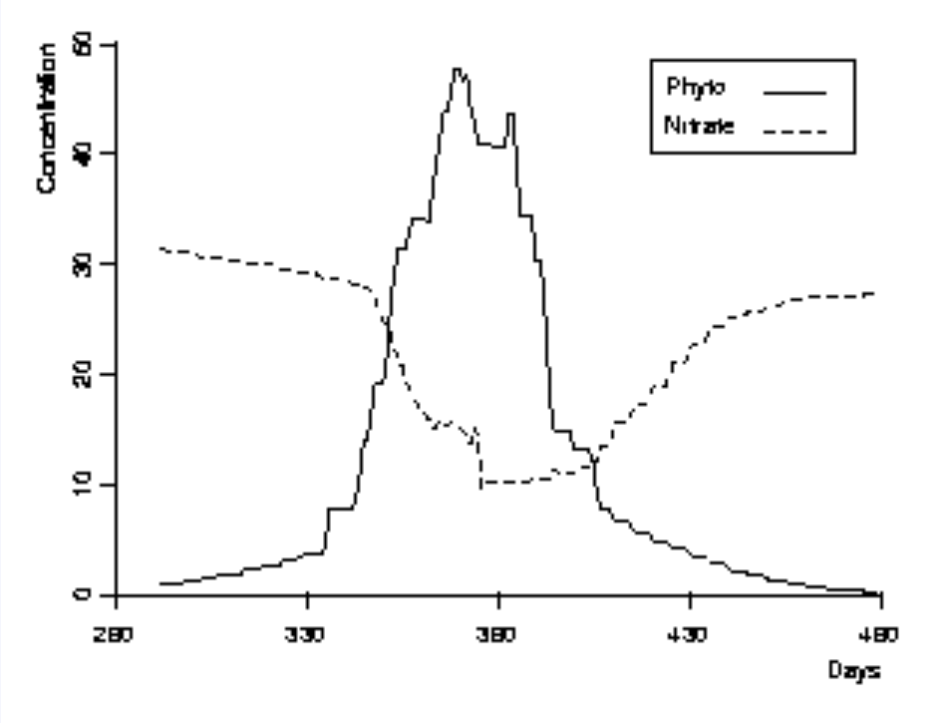
$$d[phyto,t,1] = -0.307 \times phyto - 0.495 \times zoo + 0.411 \times phyto$$

$$d[zoo,t,1] = -0.251 \times zoo + 0.615 \times 0.495 \times zoo$$

$$d[detritus,t,1] = 0.307 \times phyto + 0.251 \times zoo + 0.385 \times 0.495 \times zoo - 0.005 \times detritus$$

$$d[nitro,t,1] = -0.098 \times 0.411 \times phyto + 0.005 \times detritus$$

Processes in Ross Sea Dynamics



As phytoplankton uptakes nitrogen, its concentration increases and the nitrogen decreases. This continues until the nitrogen is exhausted, which leads to a phytoplankton die off. This produces detritus, which gradually remineralizes to replenish nitrogen. *Zooplankton grazes on phytoplankton, which slows the latter's increase and also produces detritus.*

$$d[phyto,t,1] = -0.307 \times phyto - 0.495 \times zoo + 0.411 \times phyto$$

$$d[zoo,t,1] = -0.251 \times zoo + 0.615 \times 0.495 \times zoo$$

$$d[detritus,t,1] = 0.307 \times phyto + 0.251 \times zoo + 0.385 \times 0.495 \times zoo - 0.005 \times detritus$$

$$d[nitro,t,1] = -0.098 \times 0.411 \times phyto + 0.005 \times detritus$$

A Process Model for the Ross Sea

model Ross_Sea_Ecosystem

variables: phyto, zoo, nitro, detritus
observables: phyto, nitro

process phyto_loss(phyto, detritus)
  equations: d[phyto,t,1] = −0.307 × phyto
             d[detritus,t,1] = 0.307 × phyto

process zoo_loss(zoo, detritus)
  equations: d[zoo,t,1] = −0.251 × zoo
             d[detritus,t,1] = 0.251 × zoo

process zoo_phyto_grazing(zoo, phyto, detritus)
  equations: d[zoo,t,1] = 0.615 × 0.495 × zoo
             d[detritus,t,1] = 0.385 × 0.495 × zoo
             d[phyto,t,1] = −0.495 × zoo

process nitro_uptake(phyto, nitro)
  equations: d[phyto,t,1] = 0.411 × phyto
             d[nitro,t,1] = −0.098 × 0.411 × phyto

process nitro_remineralization(nitro, detritus)
  equations: d[nitro,t,1] = 0.005 × detritus
             d[detritus,t,1] = −0.005 × detritus

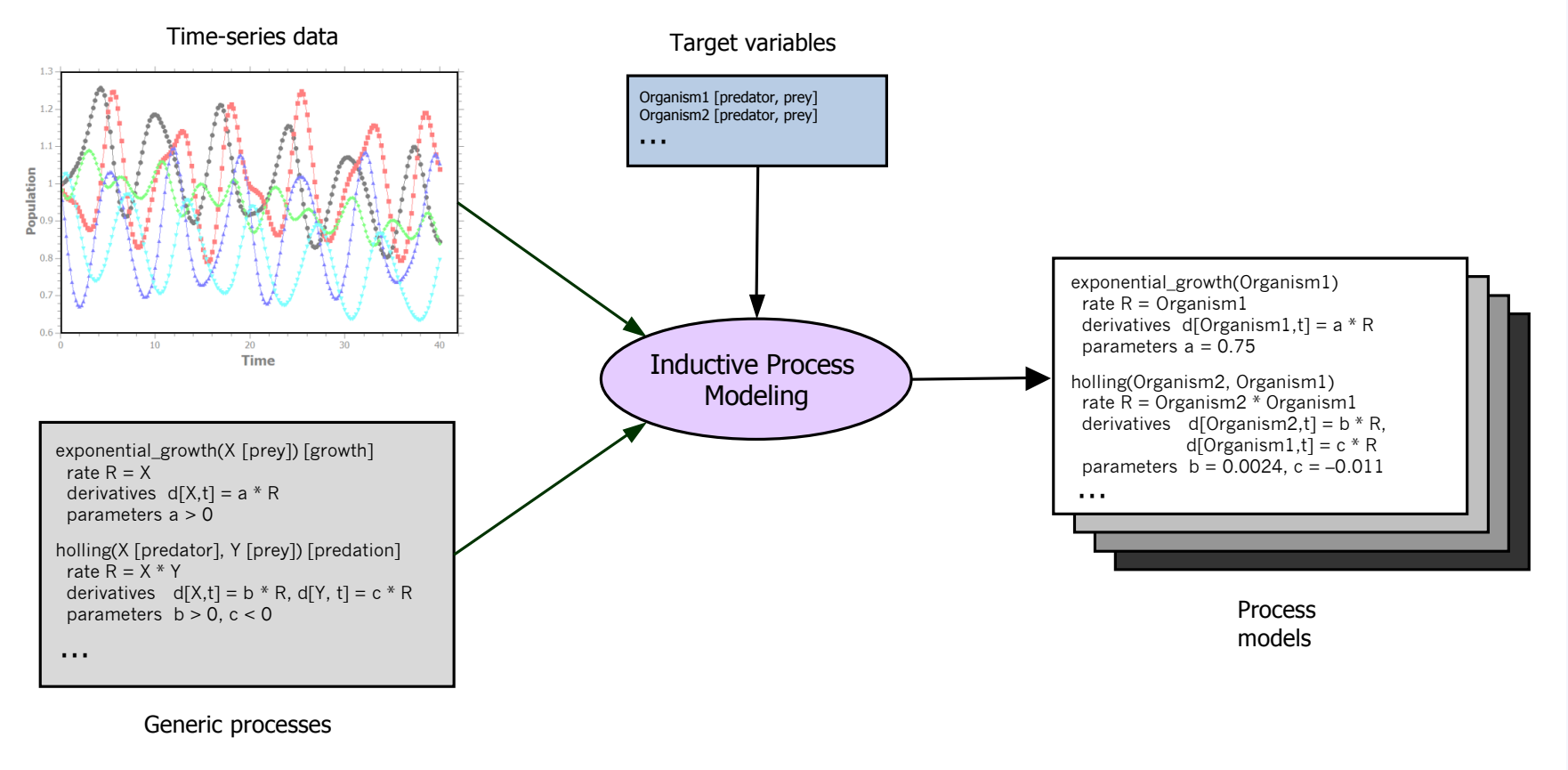
We can reformulate such an account by restating it as a *quantitative process model*.

Such a model is equivalent to a standard differential equation model, but it makes explicit assumptions about the processes involved.

Each process indicates that certain terms in equations must stand or fall together.

Inductive Process Modeling

Inductive process modeling constructs explanations of time series from background knowledge (Langley et al., *ICML-2002*).



Models are stated as sets of *differential equations* organized into higher-level *processes*.

Some Generic Processes

process exponential_loss(S, D)
  variables: S{species}, D{detritus}
  parameters: α [0, 1]
  equations: d[S, t, 1] = $-1 \times \alpha \times S$
             d[D, t, 1] = $\alpha \times S$

generic process grazing(S1, S2, D)
  variables: S1{species}, S2{species}, D{detritus}
  parameters: ρ [0, 1], γ [0, 1]
  equations: d[S1, t, 1] = $\gamma \times \rho \times S1$
             d[D ,t, 1] = $(1 - \gamma) \times \rho \times S1$
             d[S2, t, 1] = $-1 \times \rho \times S1$

generic process nutrient_uptake(S, N)
  variables: S{species}, N{nutrient}
  parameters: τ [0, ∞], β [0, 1], μ [0, 1]
  conditions: $N > \tau$
  equations: d[S, t, 1] = $\mu \times S$
             d[N, t, 1] = $-1 \times \beta \times \mu \times S$

process remineralization(N, D)
  variables: N{nutrient}, D{detritus}
  parameters: π [0, 1]
  equations:
    d[N, t, 1] = $\pi \times D$
    d[D, t, 1] = $-1 \times \pi \times D$

process constant_inflow(N)
  variables: N{nutrient}
  parameters: ν [0, 1]
  equations: d[N, t, 1] = ν

Our aquatic ecosystem library contains about 25 generic processes, including ones with alternative functional forms for loss and grazing processes.

These form the *building blocks* from which to compose models.

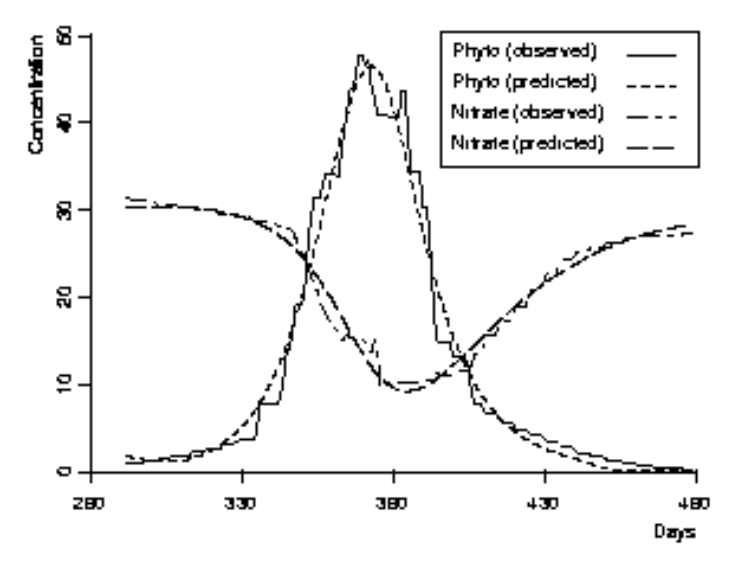
The SC-IPM System

Bridewell and Langley (2010) have reported SC-IPM, a system for inductive process modeling that:

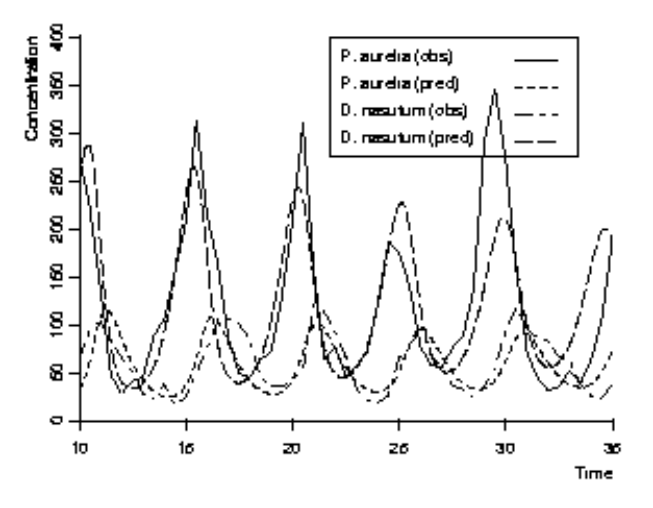
1. Uses background knowledge to generate *process instances*;
2. Combines them to produce possible *model structures*, rejecting ones that violate known *constraints*;
3. For each candidate model structure:
   a. Carries out *gradient descent search* through parameter space to find good coefficients;
   b. Invokes *random restarts* to decrease chances of local optima;
4. Returns the parameterized model with lowest squared error or a ranked list of models.

We presented encouraging results with SC-IPM on a variety of scientific data sets.

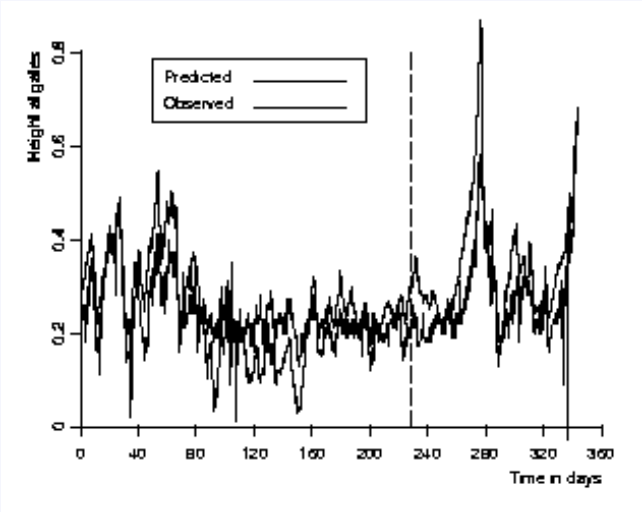
Some SC-IPM Successes



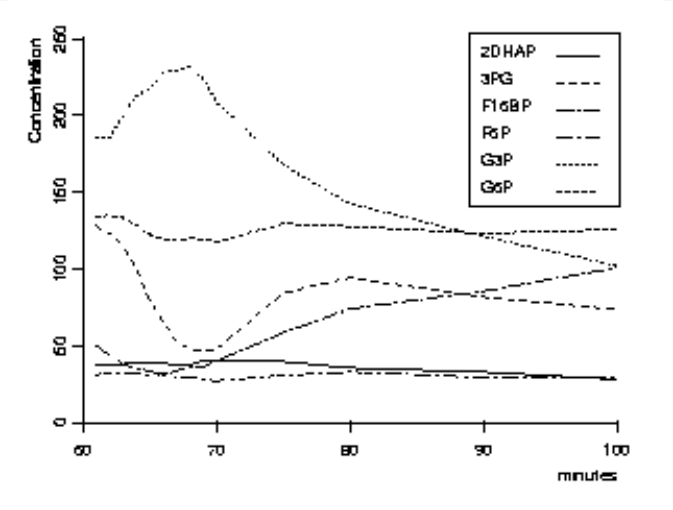
aquatic ecosystems



protist dynamics



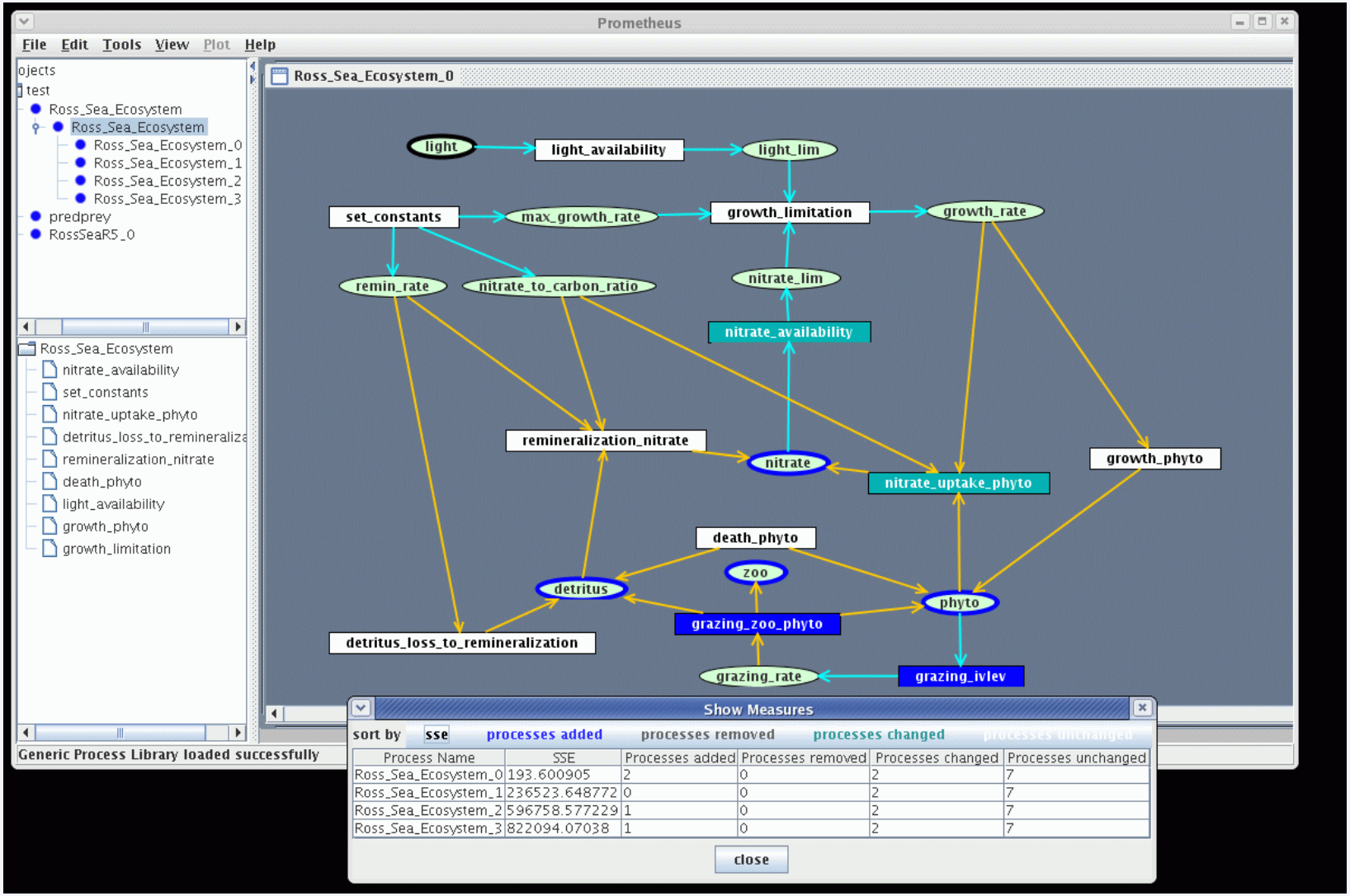
hydrology



biochemical kinetics

The PROMETHEUS System

We embedded these ideas in PROMETHEUS, an interactive system for process model construction (Bridewell et al., *IJHCS*, 2007).



Extensions to Inductive Process Modeling

In addition, we have extended the basic framework to support:

- Inductive revision of quantitative process models
  - Asgharbeygi et al. (*Ecological Modeling*, 2006)
- Hierarchical generic processes that constrain search
  - Todorovski, Bridewell, Shiran, and Langley (*AAAI-2005*)
- An ensemble-like method that mitigates overfitting effects
  - Bridewell, Bani Asadi, Langley, and Todorovski (*ICML-2005*)
- An EM-like method that estimates missing observations
  - Bridewell, Langley, Racunas, and Borrett (*ECML-2006*)

These extensions made the modeling framework more robust along a number of fronts.

Recent Progress on Process Modeling

Critiques of SC-IPM

Despite these successes, the SC-IPM system suffers from four key drawbacks, in that it:

- Evaluates *full model structures*, so disallows heuristic search
- Requires *repeated simulation* to estimate model parameters
- Invokes *random restarts* to reduce chances of local optima
- Despite these steps, it can still find poorly-fitting models

99.99 percent of CPU time

As a result, SC-IPM does not scale well to complex modeling tasks and it is not reliable.

In recent research, we have developed a new framework that avoids these problems (Langley & Arvay, *AAAI-2015*).

A New Process Formalism

SC-IPM allowed processes with only algebraic equations, only differential equations, and mixtures of them.

In our new modeling formalism, each process P must include:

- A *rate* that denotes P's speed / activation on a given time step
- An *algebraic equation* that describes P's rate as a *parameter-free* function of known variables
- One or more *derivatives* that are proportional to P's rate

This notation has important mathematical properties that assist model induction.

The new framework also comes closer to Forbus' (1984) notion of *qualitative processes*.

A Sample Process Model

Consider a process model for a simple predator-prey ecosystem:

```
exponential_growth[aurelia]
  rate         r = aurelia
  parameters   A = 0.75
  equations    d[aurelia] = A * r

exponential_loss[nasutum]
  rate         r = nasutum
  parameters   B = -0.57
  equations    d[nasutum] = B * r

holling_predation[nasutum, aurelia]
  rate         r = nasutum * aurelia
  parameters   C = 0.0024
               D = -0.011
  equations    d[nasutum] = C * r
               d[aurelia] = D * r
```

Each derivative is proportional to the algebraic rate expression.

A Sample Process Model

Consider a process model for a simple predator-prey ecosystem:

```
exponential_growth[aurelia]
  rate         r = aurelia
  parameters   A = 0.75
  equations    d[aurelia] = A * r

exponential_loss[nasutum]
  rate         r = nasutum
  parameters   B = -0.57
  equations    d[nasutum] = B * r

holling_predation[nasutum, aurelia]
  rate         r = nasutum * aurelia
  parameters   C = 0.0024
               D = -0.011
  equations    d[nasutum] = C * r
               d[aurelia] = D * r
```

This model compiles into a set of differential equations

```
d[aurelia] = 0.75 * aurelia – 0.011 * nasutum * aurelia
d[nasutum] = 0.0024 * nasutum * aurelia – 0.57 * nasutum
```

Some Generic Processes

Generic processes have a very similar but more abstract format:

```
exponential_growth(X [prey]) [growth]
  rate         r = X
  parameters   A = (> A 0.0)
  equations    d[prey] = A * r

exponential_loss(X [predator]) [loss]
  rate         r = predator
  parameters   B = (< B 0.0)
  equations    d[prey] = B * r

holling_predation(X [predator], Y [prey]) [predation]
  rate         r = X * Y
  parameters   C = (> C 0.0)
               D = (< D 0.0)
  equations    d[predator] = C * r
               d[prey] = D * r
```

As before, these are *building blocks* for constructing models.

RPM: Regression-Guided Process Modeling

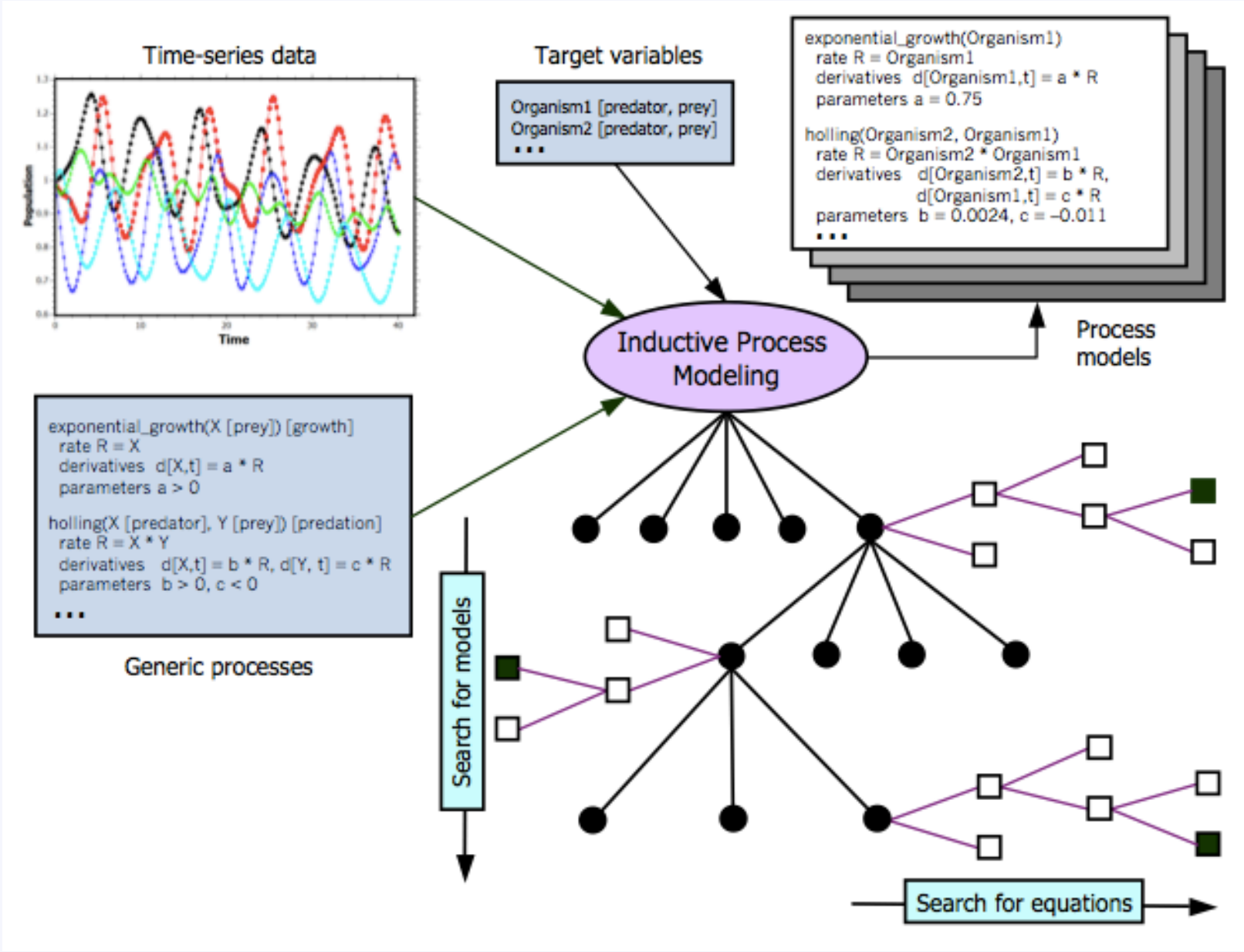
This suggests a new approach to inducing process models that our *RPM* system implements:

- Generate all process instances consistent with type constraints
- For each process P, calculate the *rate* for P on each time step
- For each dependent variable X,
  - Estimate *dX/dt* on each time step with center differencing,
  - For each subset of processes with up to *k* elements,
    - Find a regression equation for dX/dt in terms of process rates
    - If the equation's r^2 is high enough, retain for consideration
  - Add the equation with the highest r^2 to the process model

Assumes all variables observed
Rate expressions are parameter free

This approach factors the model construction task into a number of tractable components.

Two-Level Heuristic Search in RPM



Heuristics for Model Induction

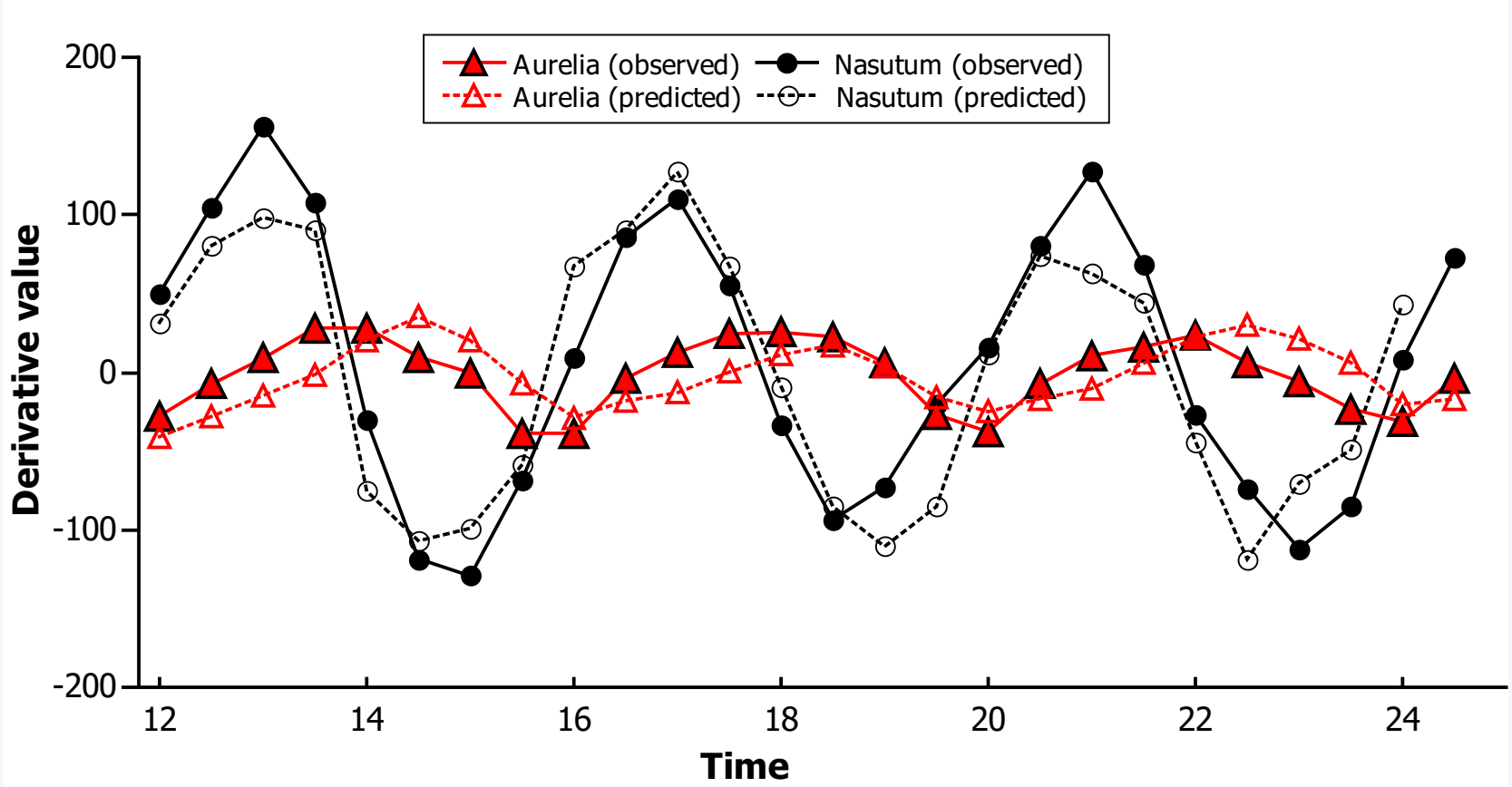
RPM uses four heuristics to guide its search through the space of process models:

- A model may include only one process instance of each type (e.g., only one variant on *predation(nasutum, aurelia)*)
- Parameters must obey numeric constraints that appear in generic forms of processes
- If an equation for one variable includes a process P, then P must appear in equations for other variables that P mentions
- Incorporate variables that participate in more processes earlier than less constrained ones

These heuristics reduce substantially the amount of search that RPM carries out during model induction.

Behavior on Natural Data

RPM matches the main trends for a simple predator-prey system.

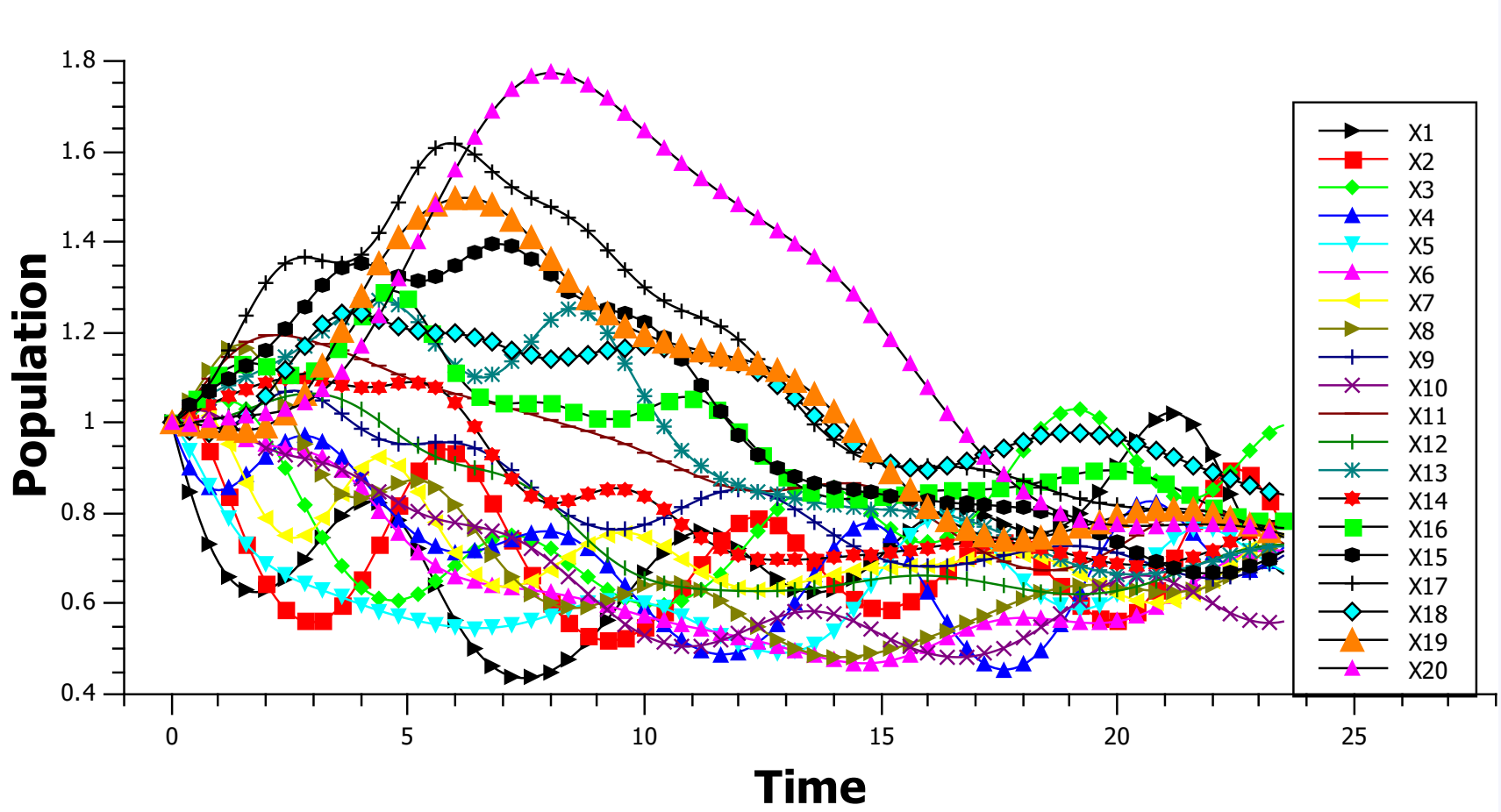


$d[aurelia] = 0.75 * aurelia - 0.11 * nasutum * aurelia \; [r^2 = 0.84]$

$d[naustum] = 0.0024 * nasutum * aurelia - 0.57 * nasutum \; [r^2 = 0.71]$

Behavior on Complex Synthetic Data

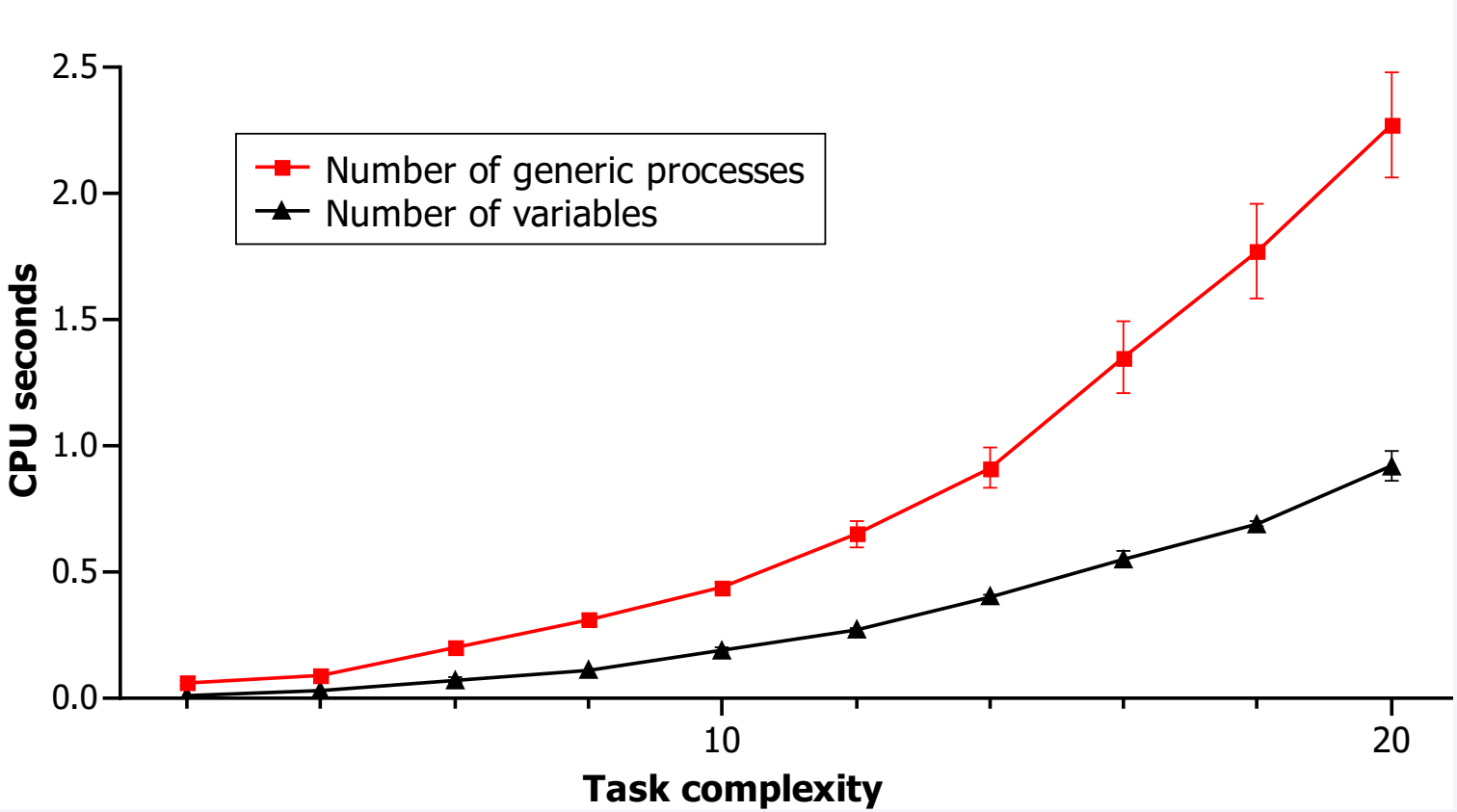
RPM also finds an accurate model for a *20-organism* food chain.



This suggests the system scales well to difficult modeling tasks.

Handling Noise and Complexity

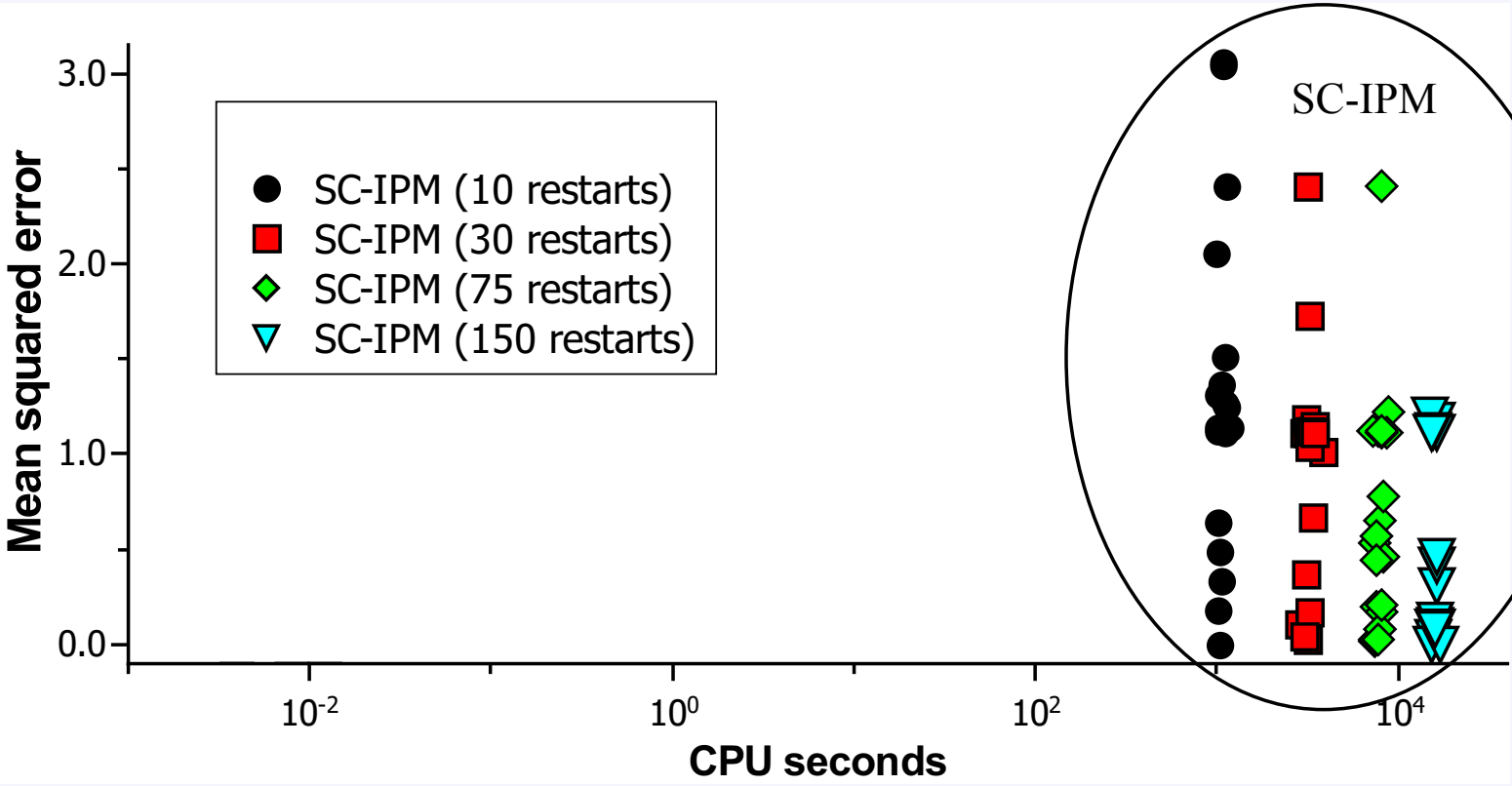
With smoothing, RPM can handle 10% noise on synthetic data.



The system also scales well to increasing numbers of generic processes and variables in the target model.

RPM and SC-IPM

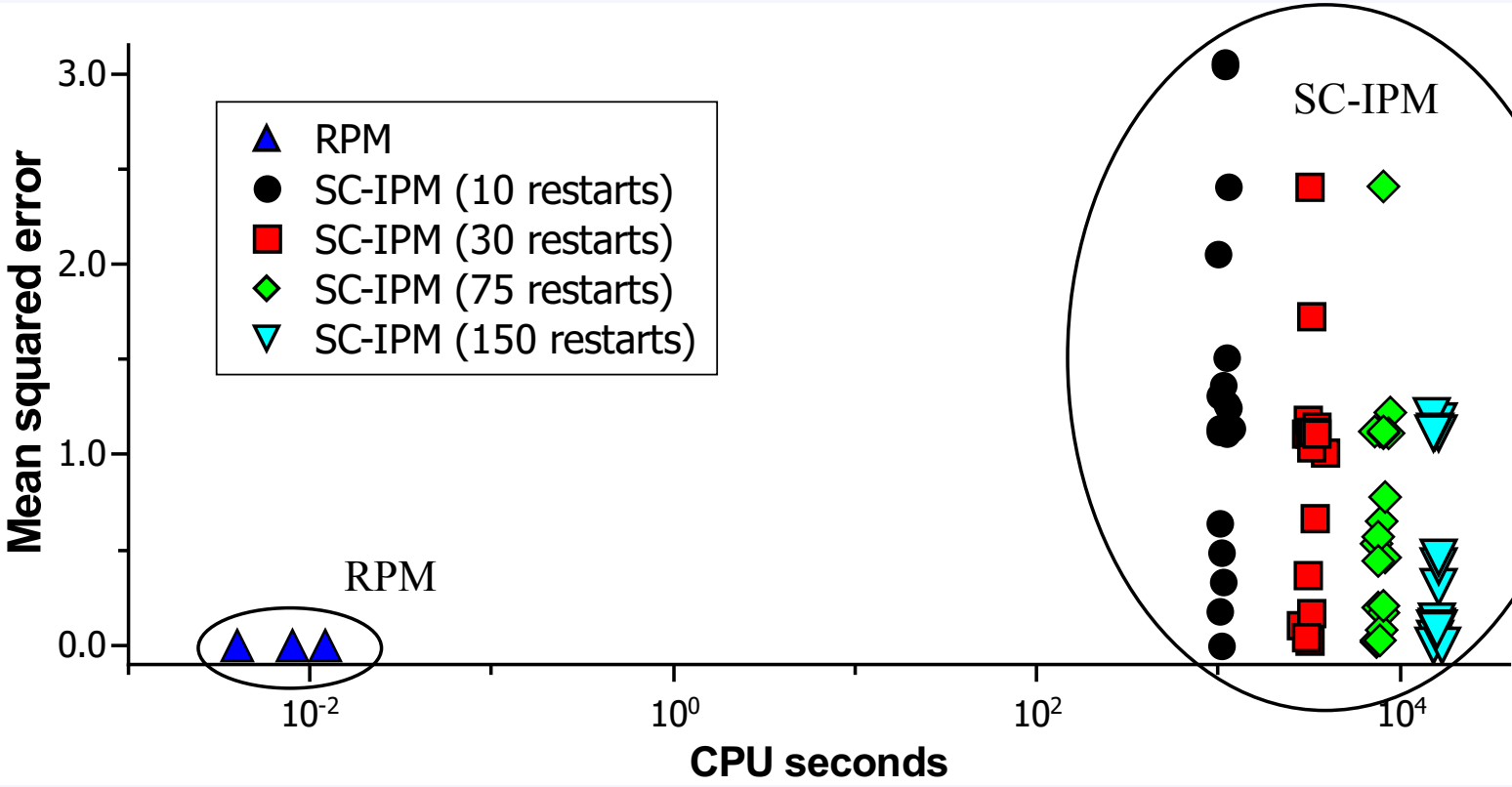
We compared RPM to SC-IPM, its predecessor, on synthetic data for a three-variable predator-prey ecosystem.



SC-IPM finds more accurate models with more restarts, but also takes longer to find them.

RPM and SC-IPM

We compared RPM to SC-IPM, its predecessor, on synthetic data for a three-variable predator-prey ecosystem.



RPM found accurate models far more reliably than SC-IPM and, at worst, ran *800,000 faster* than the earlier system.

Some Additional Extensions

Adapting Models to New Settings

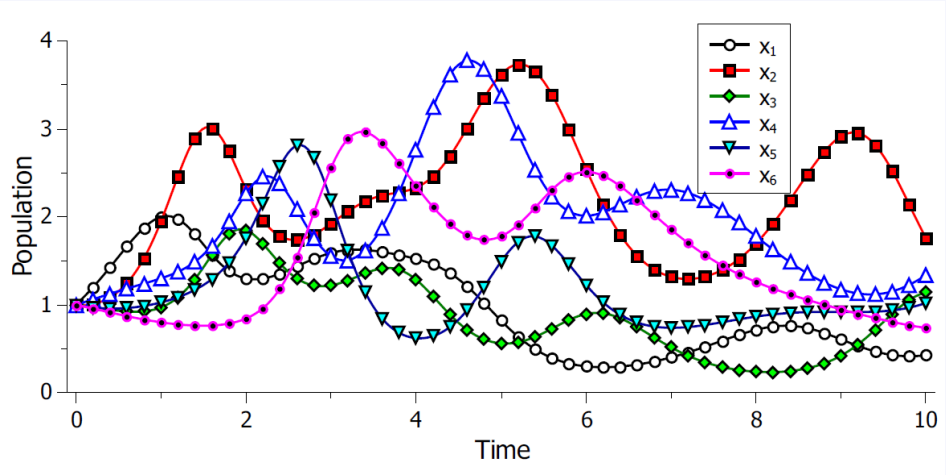
In some cases, one can adapt an existing model to observations rather inducing it from scratch.

Recent work (Arvay & Langley, ACS-2015) has extended RPM to:

- Detect anomalies / identify problematic differential equations
- Reestimate the parameters for these equations
- If necessary, remove or add processes to equations

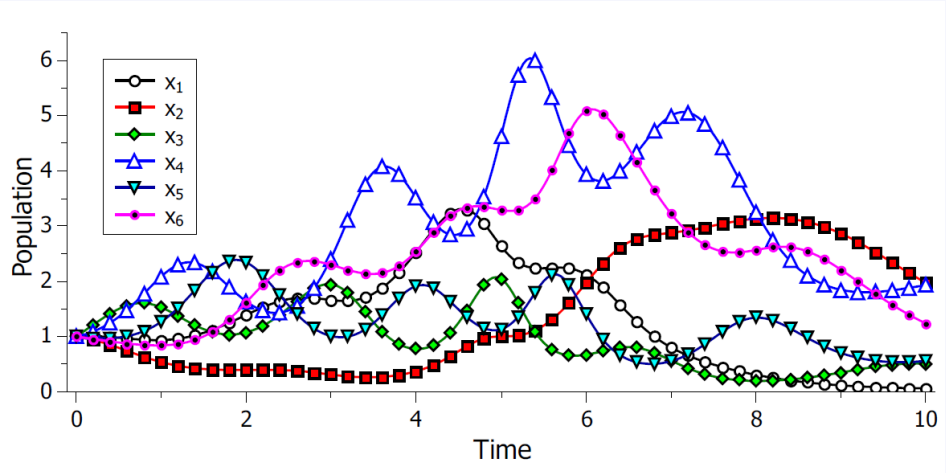
Model adaptation is appropriate when the environment changes in some ways but largely remains the same.

Effects of Environmental Changes



$$d[x1]=1.7\ x1-0.8\ x1\ x2$$
$$d[x2]=1.3\ x1\ x2-1.4\ x2\ x3$$
$$d[x3]=0.8\ x2\ x3-0.9\ x3\ x4$$
$$d[x4]=1.1\ x3\ x4-0.8\ x4\ x5$$
$$d[x5]=0.8\ x4\ x5-1.0\ x5\ x6$$
$$d[x6]=0.9\ x5\ x6-1.1\ x6$$

Initial model



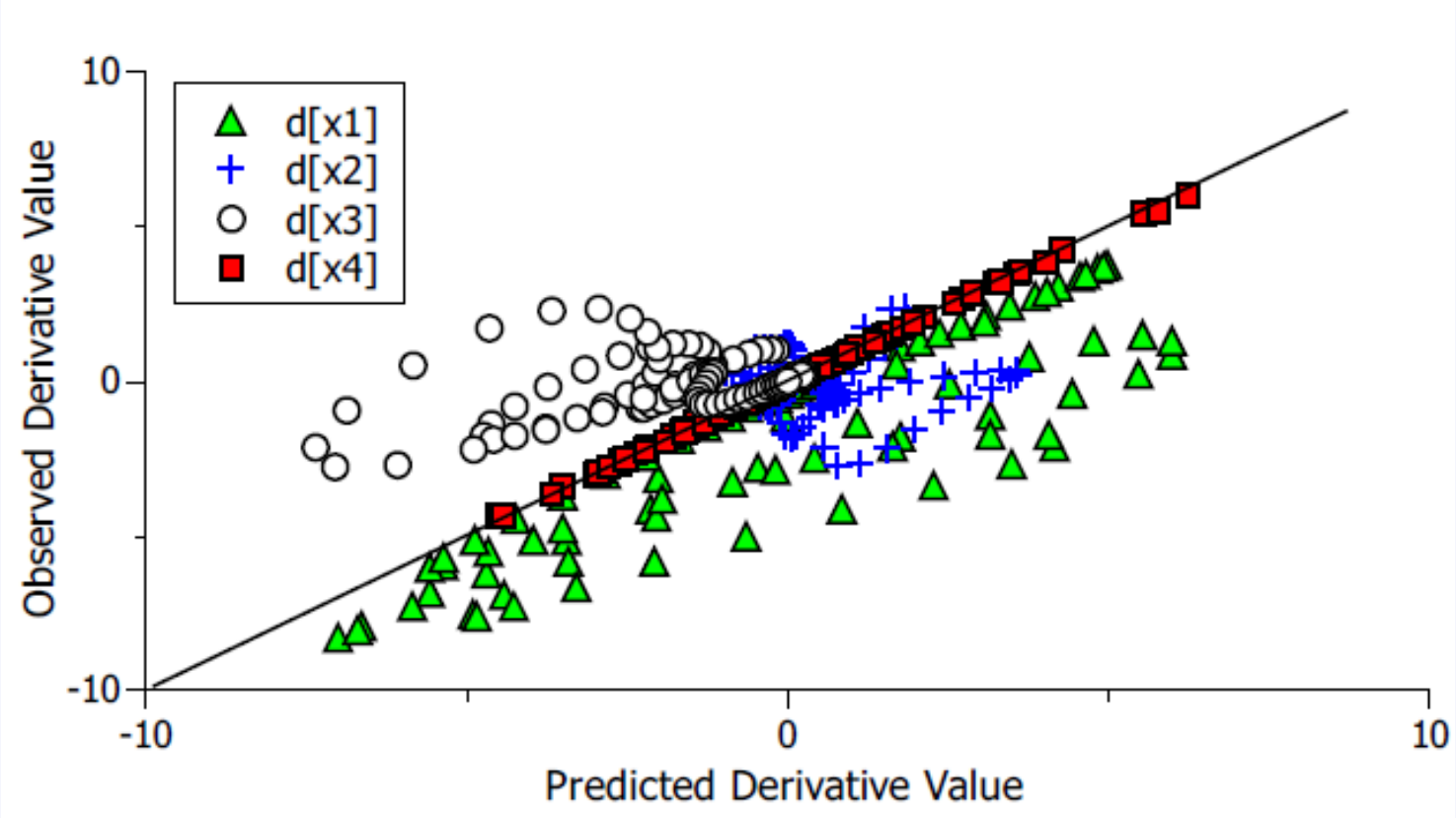
$$d[x1]=1.7\ x1-0.8\ x1\ x2-0.8\ x1\ x3$$
$$d[x2]=0.25\ x1\ x2-0.7\ x2\ x3$$
$$d[x3]=0.8\ x2\ x3-0.9\ x3\ x4+1.1\ x1\ x3$$
$$d[x4]=1.1\ x3\ x4-0.8\ x4\ x5$$
$$d[x5]=0.8\ x4\ x5-1.0\ x5\ x6$$
$$d[x6]=0.9\ x5\ x6-1.1\ x6$$

Revised model

Changes in the structure and parameters of a few equations leads to substantial changes in all trajectories.

Detecting Anomalous Derivatives

Plotting predicted derivatives against observed values lets RPM identify equations it should revise.



Here d[x4] is well predicted but other derivatives are divergent.

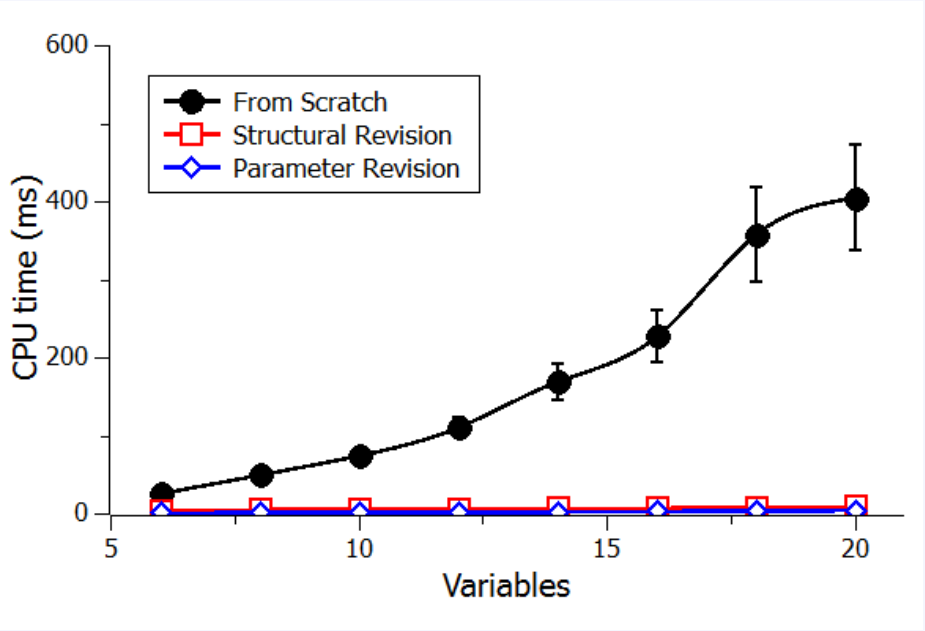
Revising a Process Model

Once RPM has identified equations that make poor predictions, it revises them by:

- Reestimating their parameters using multivariate regression
- If needed, removing / adding processes from / to each equation

The system handles each differential equation separately, but changes to earlier ones can constrain later revisions.

Studies with synthetic data show that model adaptation scales much better than induction from scratch.



Selective Induction of Process Models

In even more recent work, we have developed SPM, a system that extends RPM further by:

- Delaying binding of some variables in generic processes until it finds evidence of a relationship;
- Combining sampling of processes with backward elimination to induce more complex equations;
- Finding multiple equations for each dependent variable and then searching for ways to combine them into consistent models.

These extensions give SPM greater *coverage*, *scalability*, and *reliability* than its predecessor.

Increased Model Coverage

RPM could not induce some chemical process models because processes have the same rate; SPM avoids this problem by:

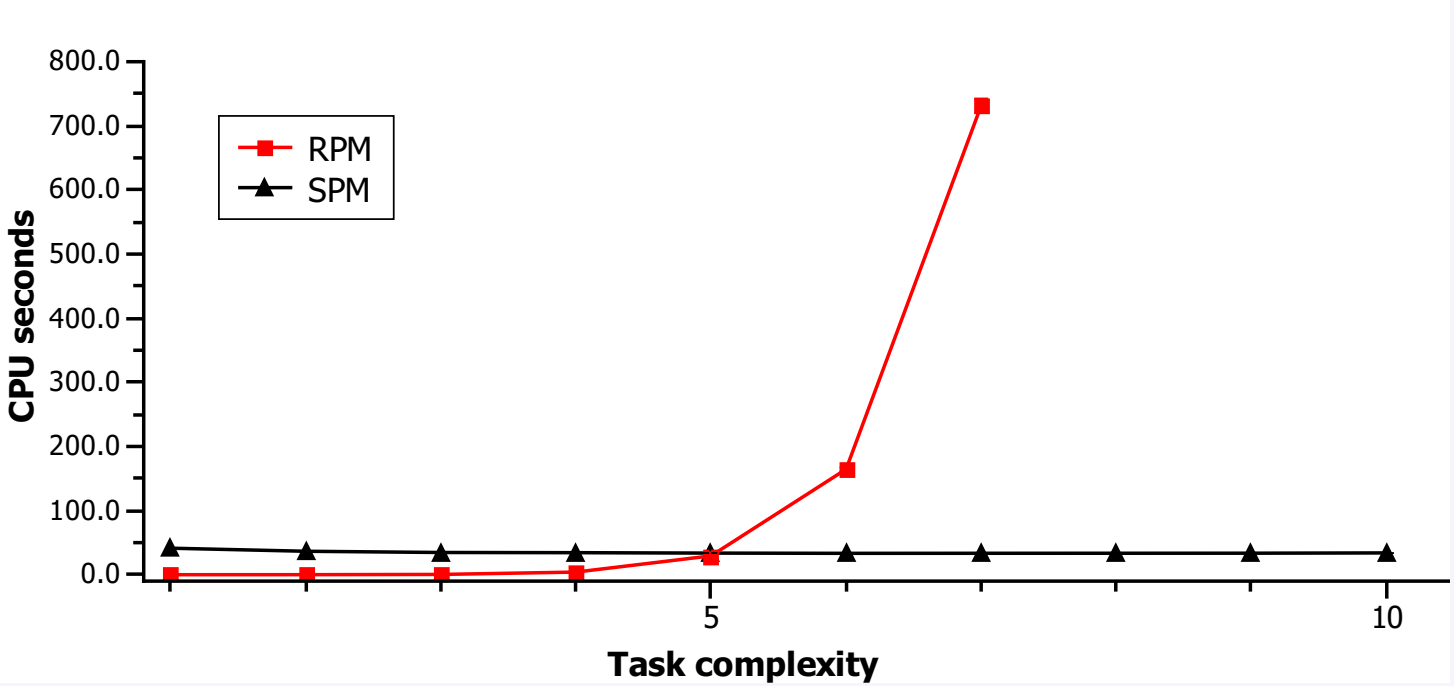
- Instantiating initially only variables in a generic process that determine its rate expression;
- Binding other variables that a process influences only when finding equations for their derivatives.

These extensions let SPM discover chemical reaction networks that RPM could not handle.

$$
\begin{aligned}
dX1/dt &= 1.1 \cdot X2 \cdot X3 - 1.6 \cdot X1 \\
dX2/dt &= 1.8 \cdot X1 - 1.5 \cdot X2 - 1.0 \cdot X2 \cdot X3 + 0.9 \cdot X5 \cdot X6 \\
dX3/dt &= 1.9 \cdot X1 + 1.1 \cdot X2 - 1.3 \cdot X3 - 1.3 \cdot X2 \cdot X3 \\
dX4/dt &= 0.9 \cdot X2 + 0.8 \cdot X3 - 2.5 \cdot X4 \cdot X5 + 0.5 \cdot X5 \cdot X6 \\
dX5/dt &= 0.9 \cdot X3 - 1.8 \cdot X4 \cdot X5 + 0.9 \cdot Z \\
dX6/dt &= 2.3 \cdot X4 \cdot X5 - 0.8 \cdot X5 \cdot X6 - 0.5 \cdot X6
\end{aligned}
$$

Better Scaling to Equation Complexity

RPM's exhaustive search for equations becomes intractable if the target involves more than five terms.



Instead, SPM combines backward elimination of rate terms with repeated sampling, giving time linear with equation complexity.

Greater Reliability of Model Induction

RPM's greedy search sometimes led it down dead ends; SPM avoids this problem by:

- Finding multiple differential equations for each target variable;
- Carrying out exhaustive depth-first search for ways to combine them into consistent models.

This strategy increased SPM's probability of inducing one or more models.

	Greedy SPM		Multi-Equation SPM	
	Percent	CPU	Percent	CPU
Nas-Aur	100	0.004±.002	100	0.004±.001
Aquatic Ecosyst	100	0.03±.012	100	0.12±.007
Predator Prey 6a	100	0.01±.003	100	0.03±.004
Predator Prey 6b	100	0.83±.004	100	2.63±.008
Predator Prey 20	100	0.81±.028	100	4.10±.100
Chemistry A	0	1.17±2.03	100	14.7±.210
Chemistry B	0	1.65±1.27	100	111.8±.610

Concluding Remarks

Related and Future Research

Our approach builds on ideas from earlier research, including:

- Qualitative representations of scientific models (Forbus, 1984)
- Inducing differential equations (Todorovski, 1995; Bradley, 2001)
- Heuristic search and multiple linear regression
- Delayed commitment and feature selection

Our plans for extending the SPM system include:

- Handling parametric rate expressions (gradient descent)
- Dealing with unobserved variables (iterative optimization)

Together, these should extend SPM's coverage and usefulness even further.

Summary Comments

Inductive process modeling is a novel and promising approach to discovering scientific models that:

- Incorporates a formalism that is familiar to many scientists
- Utilizes background knowledge about the problem domain
- Produces meaningful results from moderate amounts of data
- Generates models that explain, not just describe, observations
- Can scale well both to many processes and complex models

Although our work has focused on ecological modeling, the key ideas extend to chemistry and other domains.

For more information, see *http://www.isle.org/process/* .

eScience and Discovery Informatics

The *escience* movement champions the use of computers to aid the scientific enterprise, emphasizing two themes:

- Creation and simulation of complex explanatory models
  - E.g., differential equation models for meteorology and biology
  - However, most such models are constructed *manually*
- Collection, storage, and mining of scientific data sets
  - E.g., learned classifiers in astronomy and planetology
  - But such analyses make no contact with scientific theory

Science is about the *relation between* theory and data, and work on computational scientific discovery offers a way to join them.

This idea is central to the new field of *discovery informatics*.

Big Data and Scientific Discovery

Digital collection and storage have led to rapid growth of data in many areas.

The *big data* movement seeks to capitalize on this content, but, in science at least, must address *three* distinct issues:

- Scaling to large and heterogeneous *data sets*
- Scaling to large and complex *scientific models*
- Scaling to large *spaces of candidate models*

Handling large data sets has been widely studied and poses the fewest challenges.

We need far more work on the second two issues, for which the methods of computational scientific discovery are well suited.

Conclusion

Scientific discovery does not involve any mystical or irrational elements; we can study and even partially automate it.

Our explanation of this fascinating set of mechanisms relies on:

- Carrying out search through a space of laws or models
- Utilizing operators for generating structures and parameters
- Guiding search by data and by knowledge about the domain

Systems discover laws and models stated in the formalisms and concepts familiar to scientists.

This paradigm has already started to aid the scientific enterprise, and its importance will only grow with time.

Publications on Computational Scientific Discovery

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

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

Bridewell, W., Sanchez, J. N., Langley, P., & Billman, D. (2006). An interactive environment for the modeling and discovery of scientific knowledge. *International Journal of Human-Computer Studies*, *64*, 1099–1114.

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

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

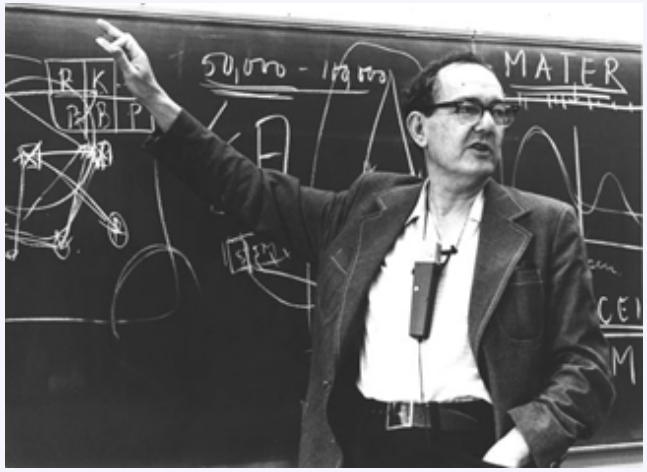
Langley, P., & Arvay, A. (2015). Heuristic induction of rate-based process models. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (pp. 537–544). Austin, TX: AAAI Press.

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

Langley, P., & Zytkow, J. M. (1989). Data-driven approaches to empirical discovery. *Artificial Intelligence*, *40*, 283–312.

In Memoriam

In 2001, the field of computational scientific discovery lost two of its founding fathers.



Herbert A. Simon
(1916 – 2001)

Jan M. Zytkow
(1945 – 2001)

Both were interdisciplinary researchers who published in computer science, psychology, philosophy, and statistics.

Herb Simon and Jan Zytkow were excellent role models for us all.

End of Presentation