Computational Discovery of Quantitative Process Models

Pat Langley

Institute for the Study of
Learning and Expertise

Center for Design Research
Stanford University

European Summer School in Artificial Intelligence
University of Ljubljana, Slovenia
July 24–28, 2023

Background and Motivation

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
- Postulated *causal chains* of interacting variables
- Hypothesized *processes* about entities' interactions

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

Quantitative Explanatory Models

There has been substantial research on computational discovery that has addressed either:

- Inducing numeric laws that describe *quantitative* observations
- Abducing structural accounts to explain *qualitative* phenomena

But scientists in advanced fields often combine both activities to create models that:

- Postulate unobserved structural relations among entities
- Incorporate functional forms with numeric parameters

Can we also develop systems that discover such *quantitative explanatory models*?

Constructing Quantitative Explanations

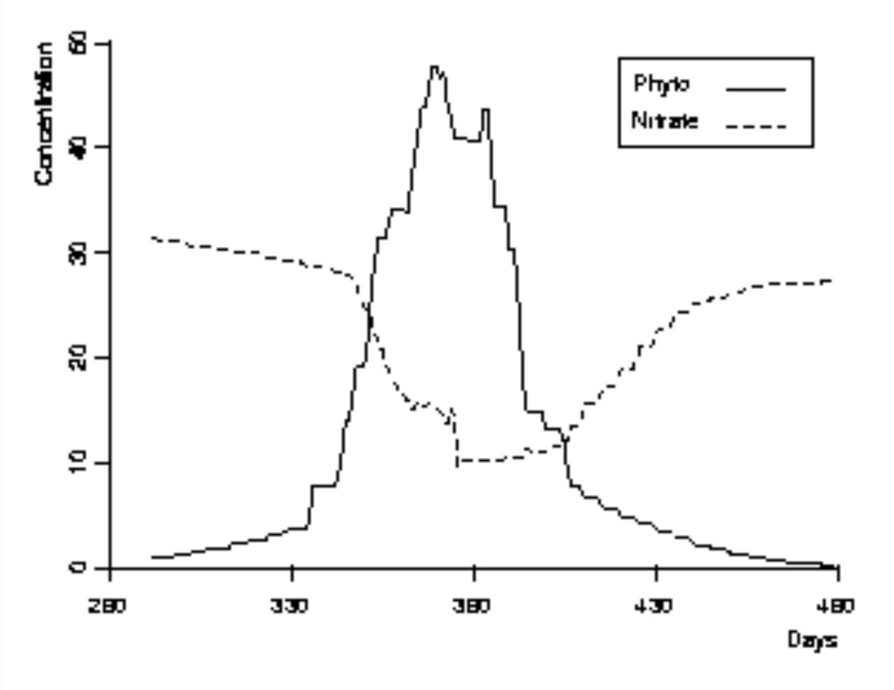
We have seen some research on the computational discovery of quantitative explanations:

- Inferring *abstract causal models* / structural equation models (Glymour et al., 1987; Spirtes et al., 1993)
- Identifying sets of *linked differential equations* (Dzeroski & Todorovski, 1993; Stolle & Bradley, 1998; Koza et al., 2001)

These combined distinct numeric equations into qualitative structures, but they remained reasonably shallow.

Can we also automate the discovery of quantitative models that postulate *unobserved variables and processes*?

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.

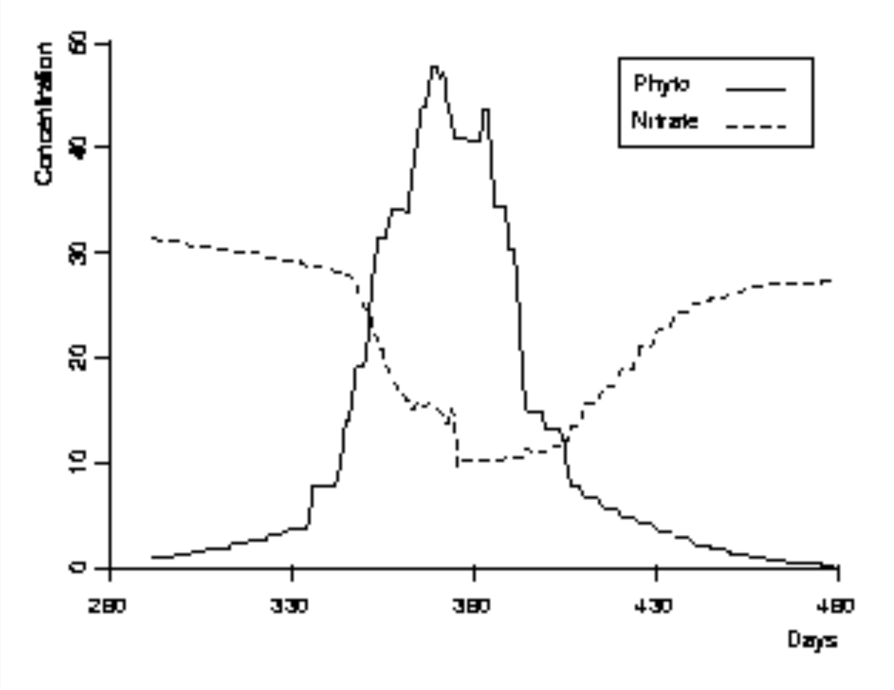
d[phyto,t,1] = – 0.307 × phyto – 0.495 × zoo + 0.411 × phyto

d[zoo,t,1] = – 0.251 × zoo + 0.615 × 0.495 × zoo

d[detritus,t,1] = 0.307 × phyto + 0.251 × zoo + 0.385 × 0.495 × zoo – 0.005 × detritus

d[nitro,t,1] = – 0.098 × 0.411 × phyto + 0.005 × 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.

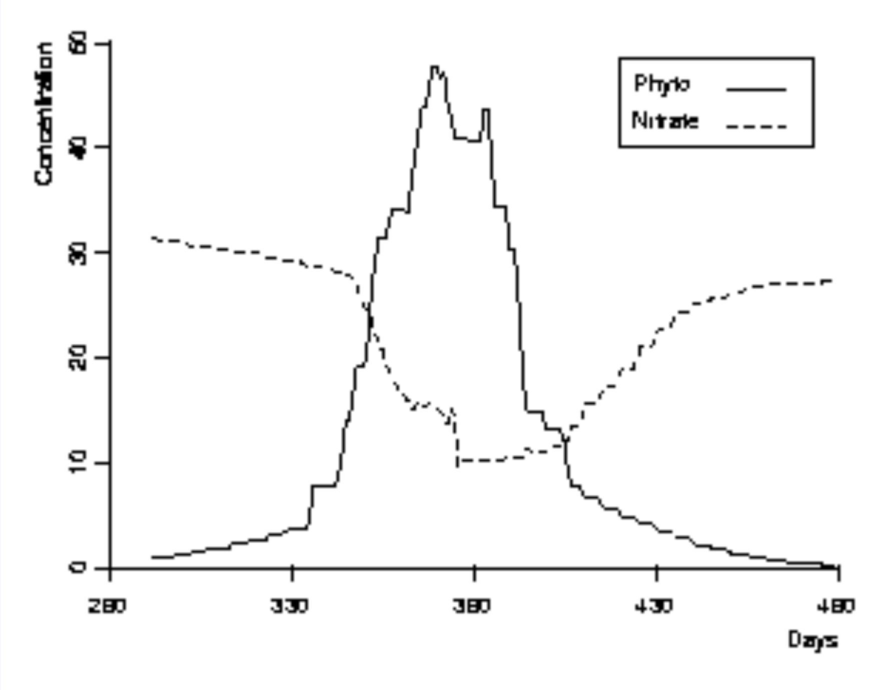
d[phyto,t,1] = – 0.307 × phyto – 0.495 × zoo + 0.411 × phyto

d[zoo,t,1] = – 0.251 × zoo + 0.615 × 0.495 × zoo

d[detritus,t,1] = 0.307 × phyto + 0.251 × zoo + 0.385 × 0.495 × zoo – 0.005 × detritus

d[nitro,t,1] = – 0.098 × 0.411 × phyto + 0.005 × 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.

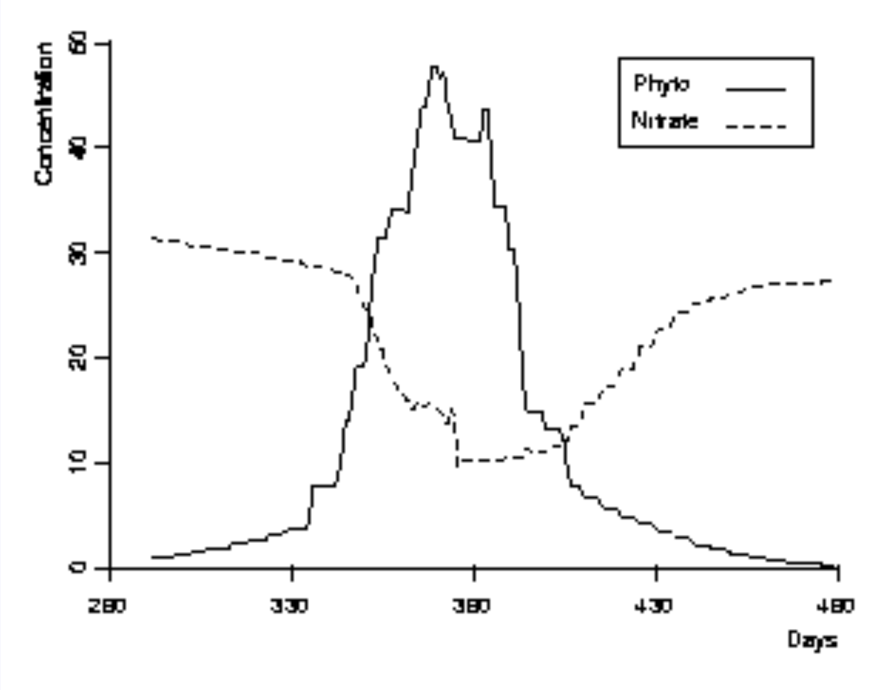
$$\boxed{\text{d[phyto,t,1]} =} - 0.307 \times \text{phyto} - 0.495 \times \text{zoo} + \boxed{0.411 \times \text{phyto}}$$

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

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

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

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.*

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

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

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

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

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
```

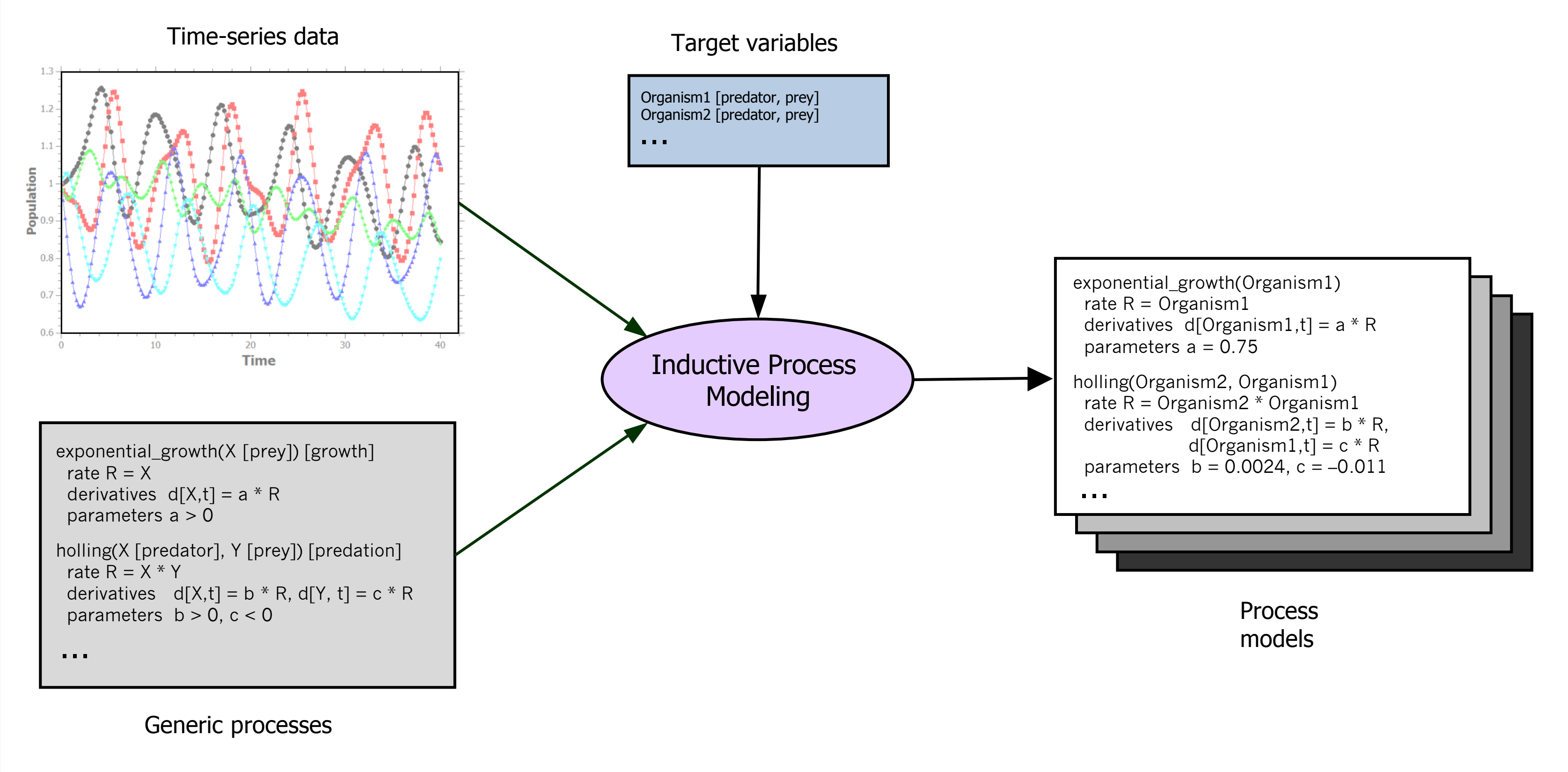
We can reformulate such an account by restating it as a *quantitative process model*.

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*).



Time-series data

Target variables

```
Organism1 [predator, prey]
Organism2 [predator, prey]
. . .
```

Inductive Process Modeling

Generic processes

```
exponential_growth(X [prey]) [growth]
  rate R = X
  derivatives  d[X,t] = a * R
  parameters a > 0

holling(X [predator], Y [prey]) [predation]
  rate R = X * Y
  derivatives   d[X,t] = b * R, d[Y, t] = c * R
  parameters  b > 0, c < 0

. . .
```

Process models

```
exponential_growth(Organism1)
  rate R = Organism1
  derivatives  d[Organism1,t] = a * R
  parameters a = 0.75

holling(Organism2, Organism1)
  rate R = Organism2 * Organism1
  derivatives   d[Organism2,t] = b * R,
                d[Organism1,t] = c * R
  parameters  b = 0.0024, c = –0.011
  . . .
```

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: $\alpha$ [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: $\rho$ [0, 1], $\gamma$ [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: $\tau$ [0, $\infty$], $\beta$ [0, 1], $\mu$ [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: $\pi$ [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: $\nu$ [0, 1]
  equations:   $d[N, t, 1] = \nu$
```

Our aquatic ecosystem library contained 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.

Searching the Space of Model Structures

We developed multiple 'IPM' systems that induce process models from generic components in four stages:

1. Instantiate known generic processes with specific entities, subject to type specifications;
2. Combine these instantiated processes into candidate model structures, rejecting disconnected structures;
3. For each model structure, carry out search through parameter space to find good coefficients;
4. Return the parameterized model with the best overall score (e.g., lowest squared error).

We have reported variants on this approach in numerous papers (Bridewell et al., *MLj*, 2008; Bridewell & Langley, *TopiCS*, 2010).

Searching the Space of Model Parameters

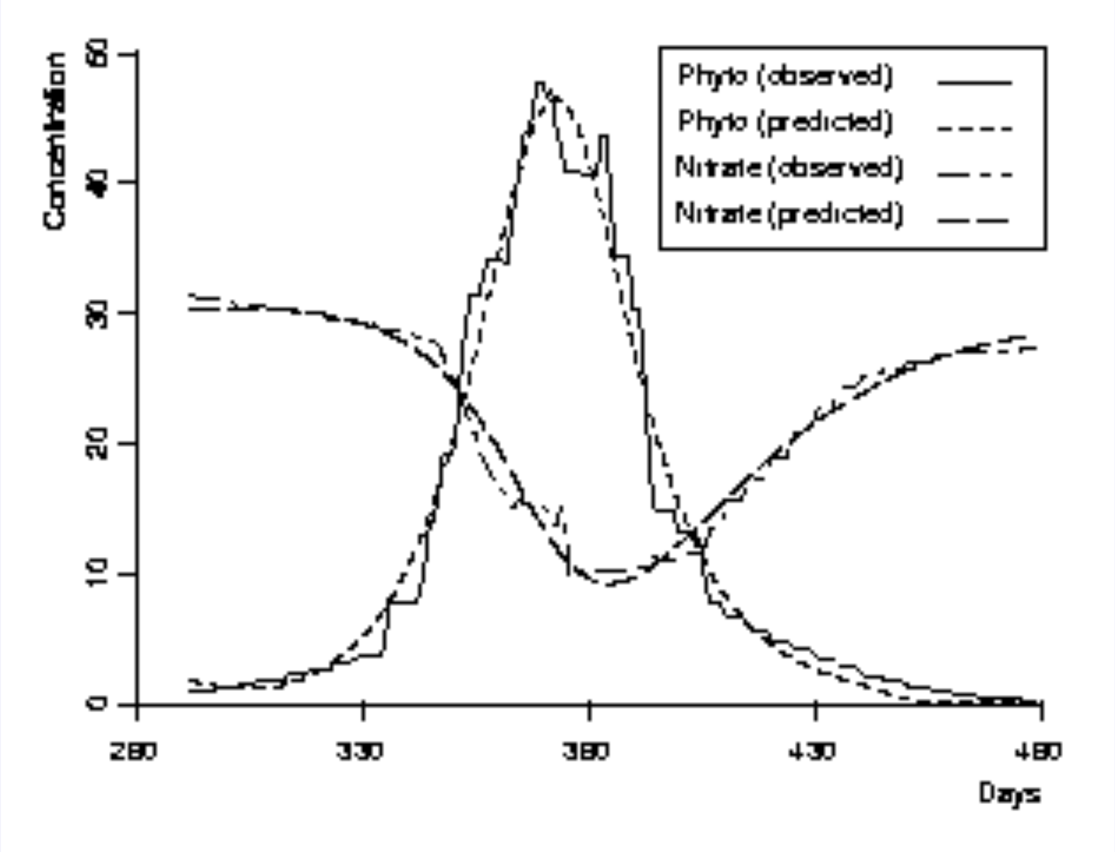
To estimate the parameters for each generic model structure, our induction algorithms:

1. Selected random initial values that fall within ranges specified in the generic processes;
2. Improved these parameters using a conjugate gradient method until it reaches a local optimum;
3. Repeated the process N times and selected the best-scoring set of parameter values.

This multi-level method gave reasonable fits to time-series data for some domains, but it was computationally intensive.

Each step in the gradient descent required simulating the model’s trajectory to calculate its error.

Results on Training Data from Ross Sea

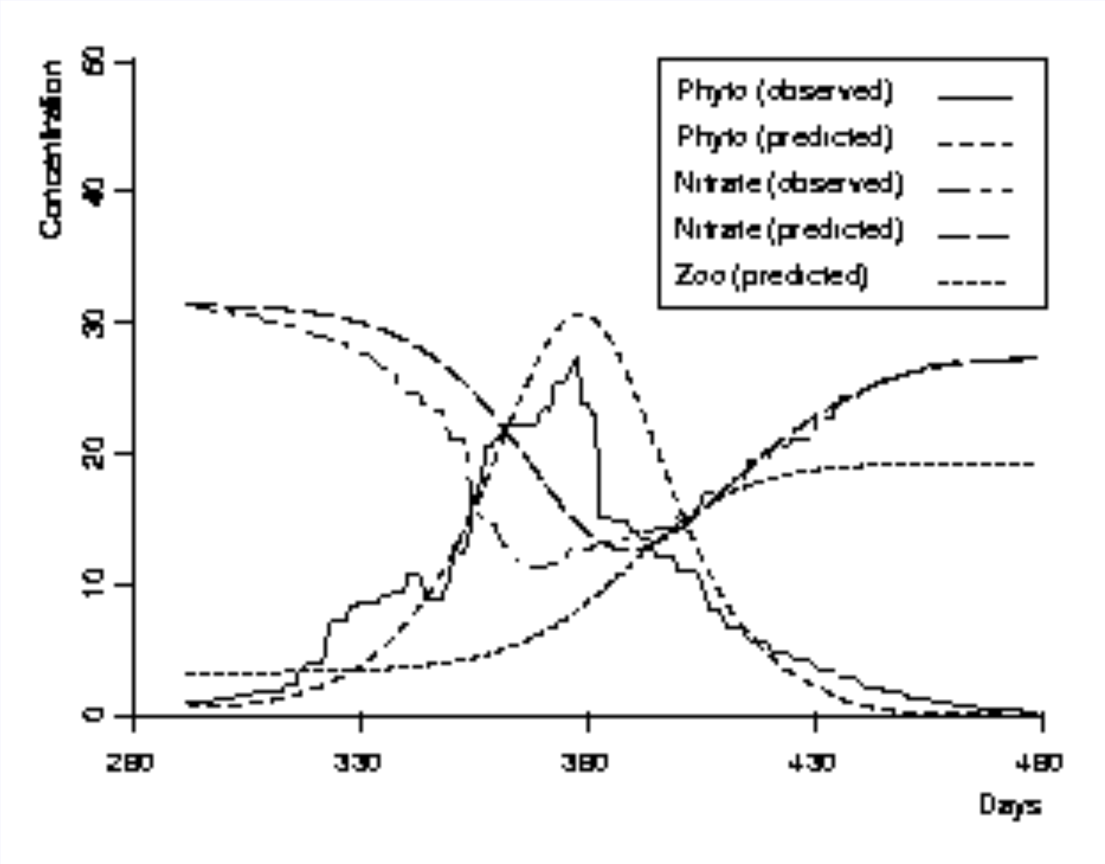


We provided IPM with 188 samples of phytoplankton, nitrate, and ice measures taken from the Ross Sea.

From 2035 distinct model structures, it found accurate models that limited phyto growth by the nitrate and the light available.

Some high-ranking models incorporated zooplankton, whereas others did not.

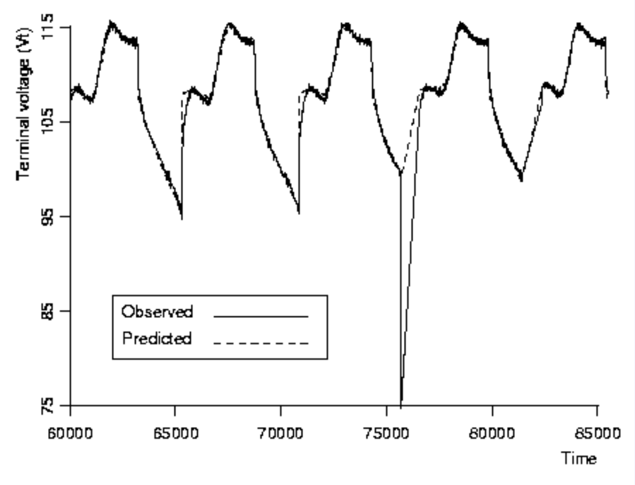
Results on Test Data from Ross Sea



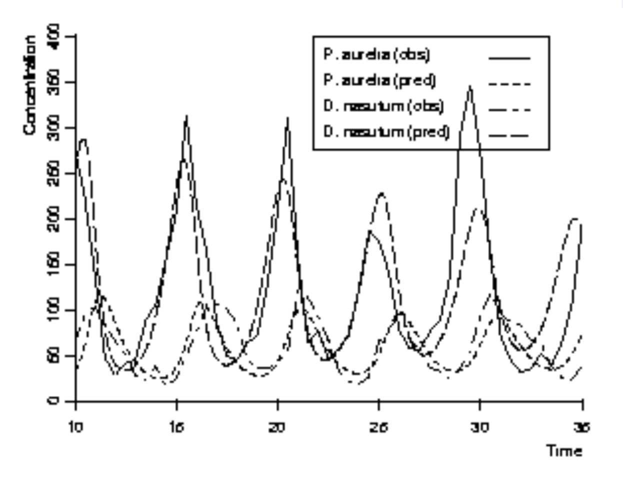
Generalization to a second year’s data benefited from treating initial zooplankton concentration as a free model parameter.

Another good-fitting model suggested that the nitrogen to carbon ratio varies as a function of available light.

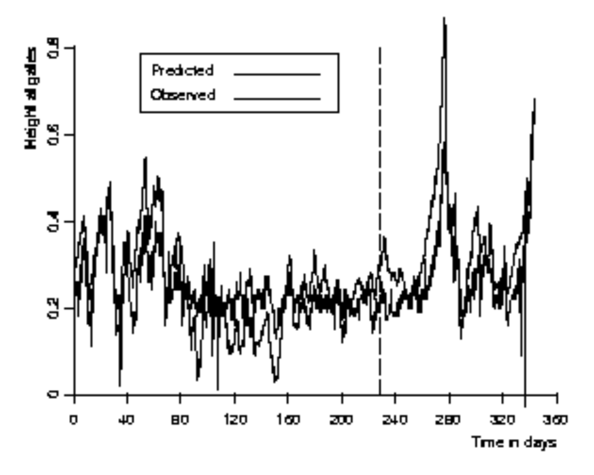
Other Results with Process Modeling



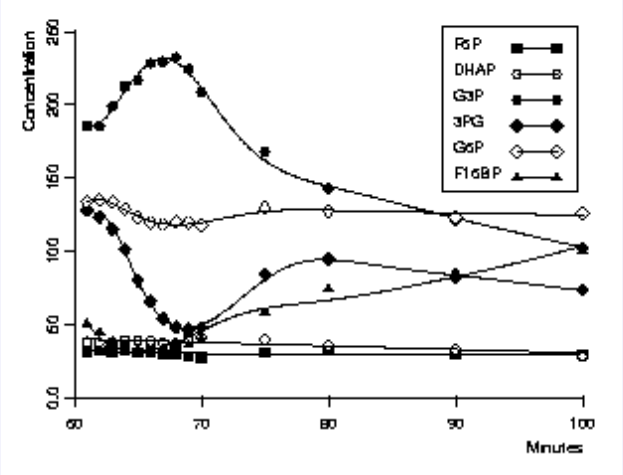
power systems



protist dynamics



hydrology



biochemical kinetics

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.

Interfacing with Scientists

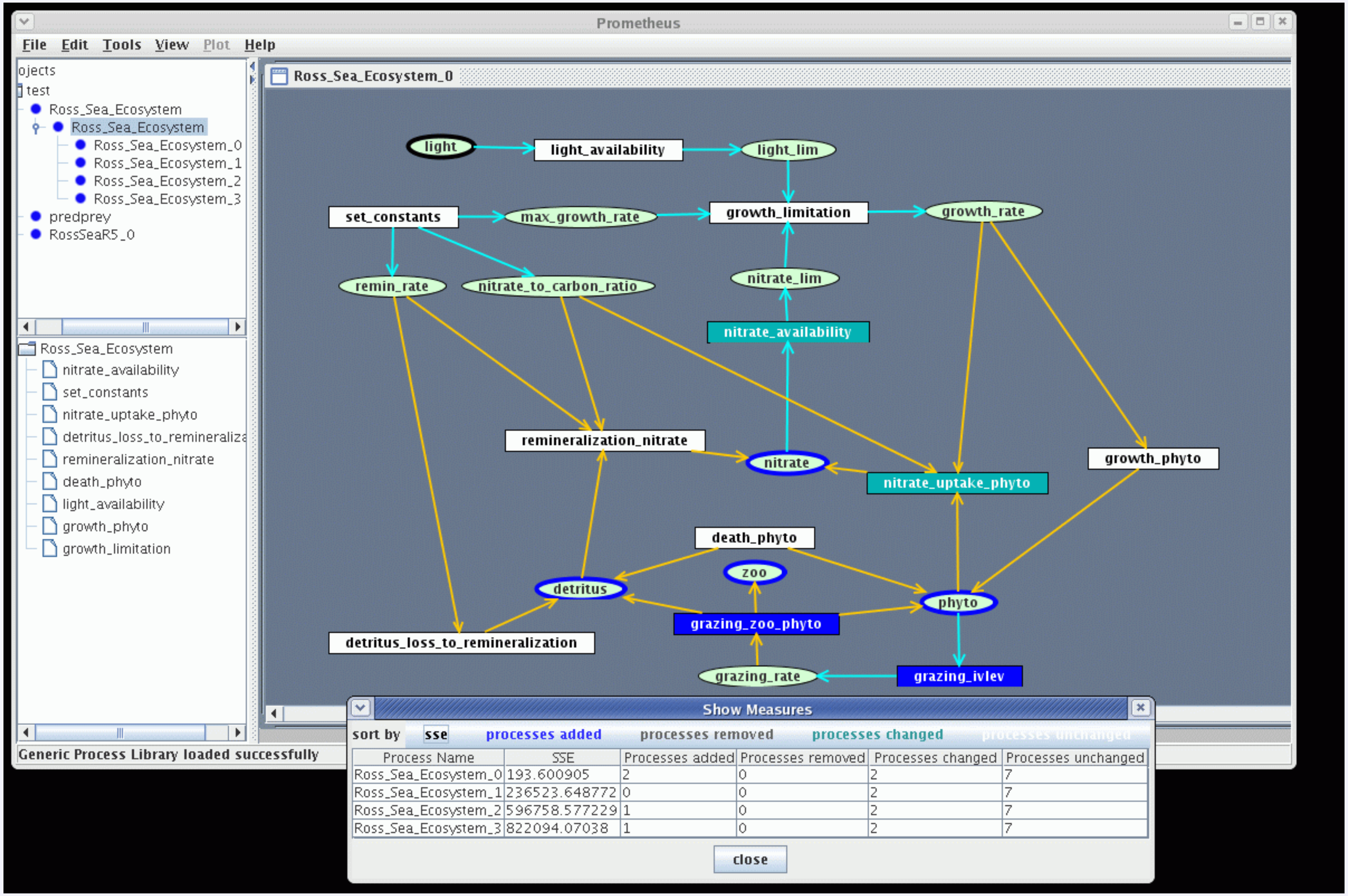
Because few scientists want to be replaced, we also developed an interactive environment, PROMETHEUS, that lets users:

- Specify a quantitative process model of the target system;
- Display and edit the model's structure and details graphically;
- Simulate the model's behavior over time and situations;
- Compare the model's predicted behavior to observations;
- Invoke a revision module in response to detected anomalies.

The environment offers computational assistance in forming and evaluating models but lets the user retain control.

The PROMETHEUS System

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



Generic Process Library loaded successfully

Process Name	SSE	Processes added	Processes removed	Processes changed	Processes unchanged
Ross_Sea_Ecosystem_0	193.600905	2	0	2	7
Ross_Sea_Ecosystem_1	236523.648772	0	0	2	7
Ross_Sea_Ecosystem_2	596758.577229	1	0	2	7
Ross_Sea_Ecosystem_3	822094.07038	1	0	2	7

Constraint-Guided Process Modeling

Knowledge and Search in Discovery

Traditional treatments of problem solving hold that knowledge reduces the amount of search.

- But adding generic processes leads to a combinatorial *increase* in the number of candidate structures.

Yet scientists are not overwhelmed by the size of model spaces and they reject many structures as unacceptable.

This suggests *two* forms of scientific background knowledge:

- *components* used to generate candidate model structures
- *constraints* on allowable combinations of such components

This distinction seldom occurs in the literature, but it appears key to understanding scientific explanation.

Constraints on Ecosystem Models

Our discussions with ecologists confirmed that constraints play an important role in model acceptability.

Some plausible constraints for models of ecosystems include:

- There must be at most one growth process for each species.
- A limited growth process cannot occur without a nutrient limitation process and vice versa.
- There must be no more than one predation process between any two species.

We have developed a formal notation that lets our systems use such constraints during inductive process modeling.

Inducing Process Models with Constraints

Our extended framework for the discovery of process models:

- Encoded modular constraints on process combinations
- Used these constraints to eliminate unacceptable models
- Reduced search through the model space, which
  - Led to far more efficient model construction
  - Produced little or no increase in generalization error
  - Improved the comprehensibility of generated models

The resulting systems were more robust in their ability to induce process models (Bridewell & Langley, *TopiCS*, 2010).

The SC-IPM System

Bridewell and Langley’s (2010) SC-IPM system incorporated these ideas in that it:

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

Experiments with SC-IPM produced far more reliable results.

Discovering Model Constraints

In related work, Todorovski et al. (AAAI-2012) reported another system that:

- Used inductive process modeling to generate a set of models;
- Separated these into accurate and inaccurate model structures;
- Described each model structure in terms of relational literals;
- Learned relational rules that can distinguish the two classes;
- Transformed the rules into constraints on model structures; and
- Used these constraints to guide search on future modeling tasks.

Experiments revealed this produced a tenfold speedup on novel modeling tasks with little or no loss in accuracy.

Rate-Based 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

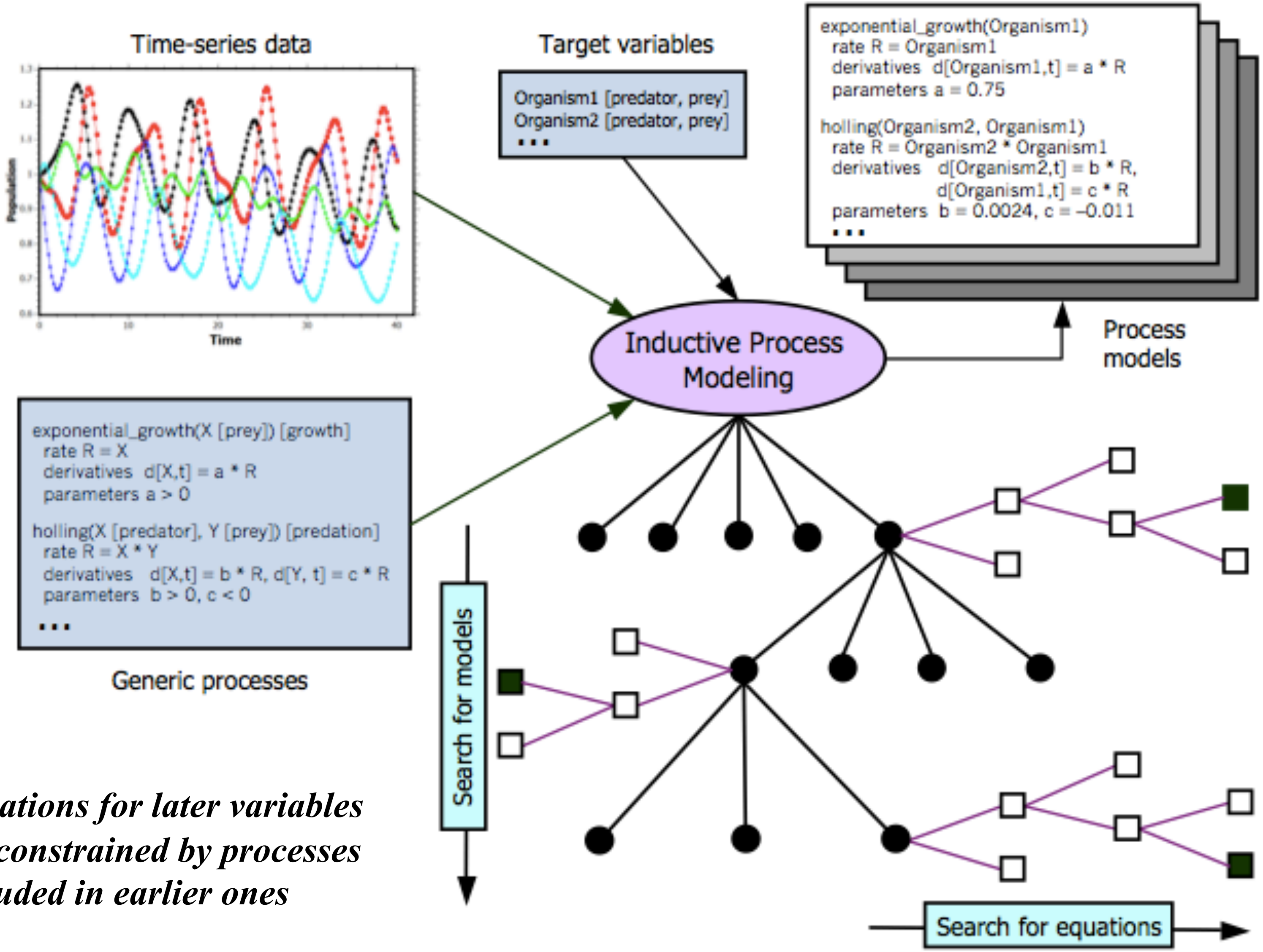
This suggests a new approach to inducing process models that our *RPM* system implements:

- 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,
  - Find a regression equation for dX/dt in terms of process rates
  - If r^2 for equation is high enough, add it 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



```
Organism1 [predator, prey]
Organism2 [predator, prey]
...
```

```
exponential_growth(Organism1)
  rate R = Organism1
  derivatives  d[Organism1,t] = a * R
  parameters a = 0.75

holling(Organism2, Organism1)
  rate R = Organism2 * Organism1
  derivatives   d[Organism2,t] = b * R,
                d[Organism1,t] = c * R
  parameters  b = 0.0024, c = –0.011
  ...
```

```
exponential_growth(X [prey]) [growth]
  rate R = X
  derivatives  d[X,t] = a * R
  parameters a > 0

holling(X [predator], Y [prey]) [predation]
  rate R = X * Y
  derivatives   d[X,t] = b * R, d[Y, t] = c * R
  parameters  b > 0, c < 0
  ...
```

Equations for later variables are constrained by processes included in earlier ones

Heuristics for Model Induction

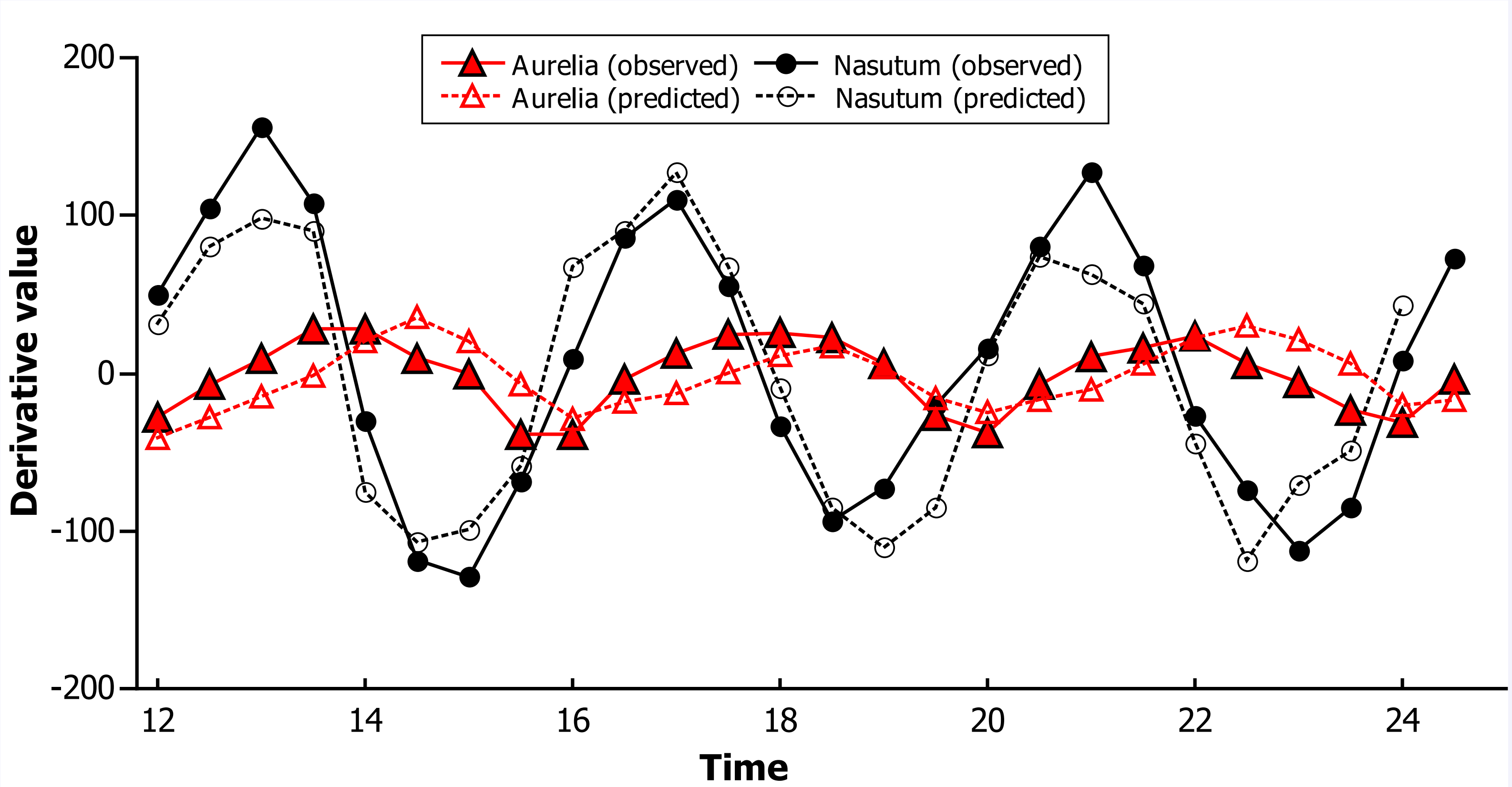
RPM uses four heuristics to guide its search through the space of process models:

- 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.

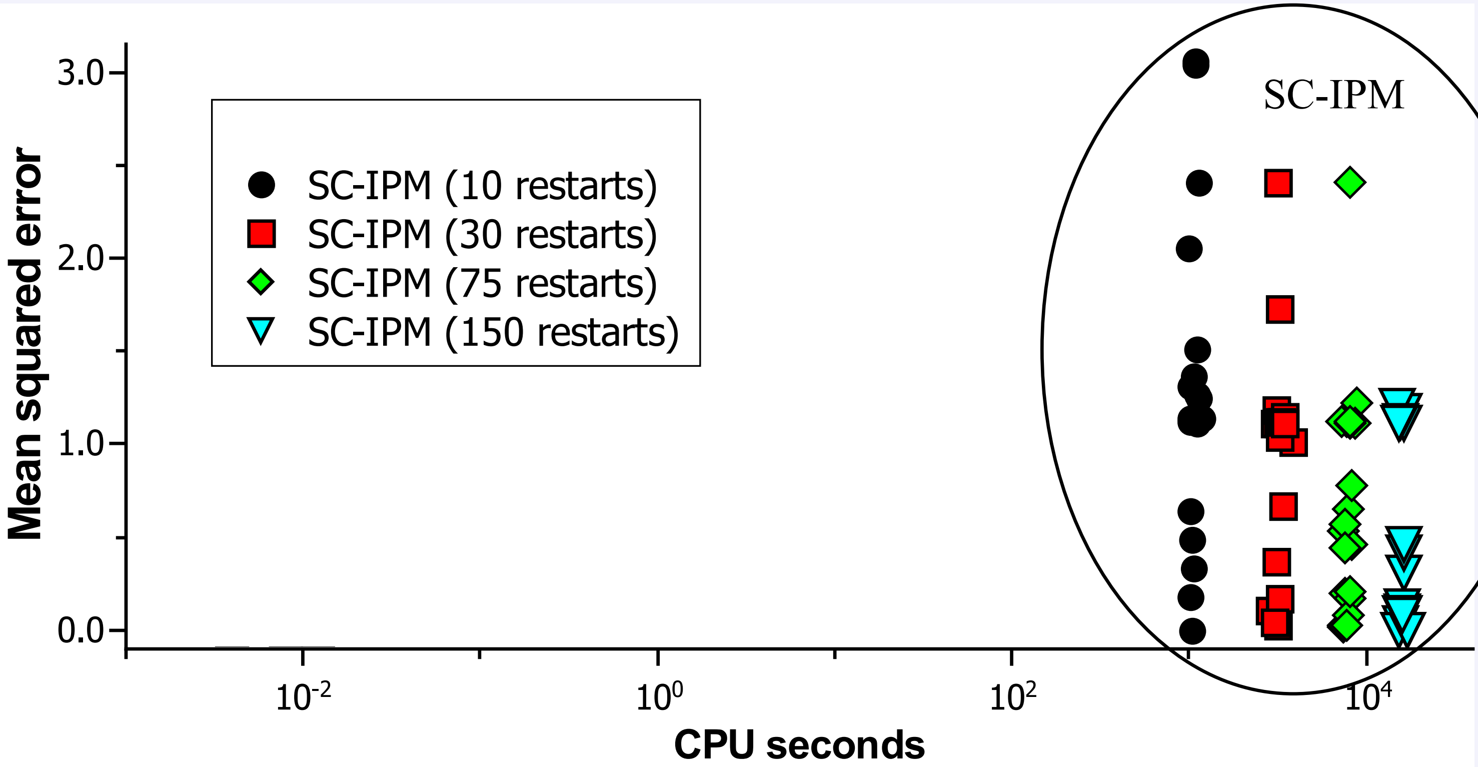


*d[aurelia] = 0.75 * aurelia − 0.11 * nasutum * aurelia [r*^{2} = 0.84]

*d[naustum] = 0.0024 * nasutum * aurelia − 0.57 * nasutum [r^2 = 0.71]*

RPM and SC-IPM

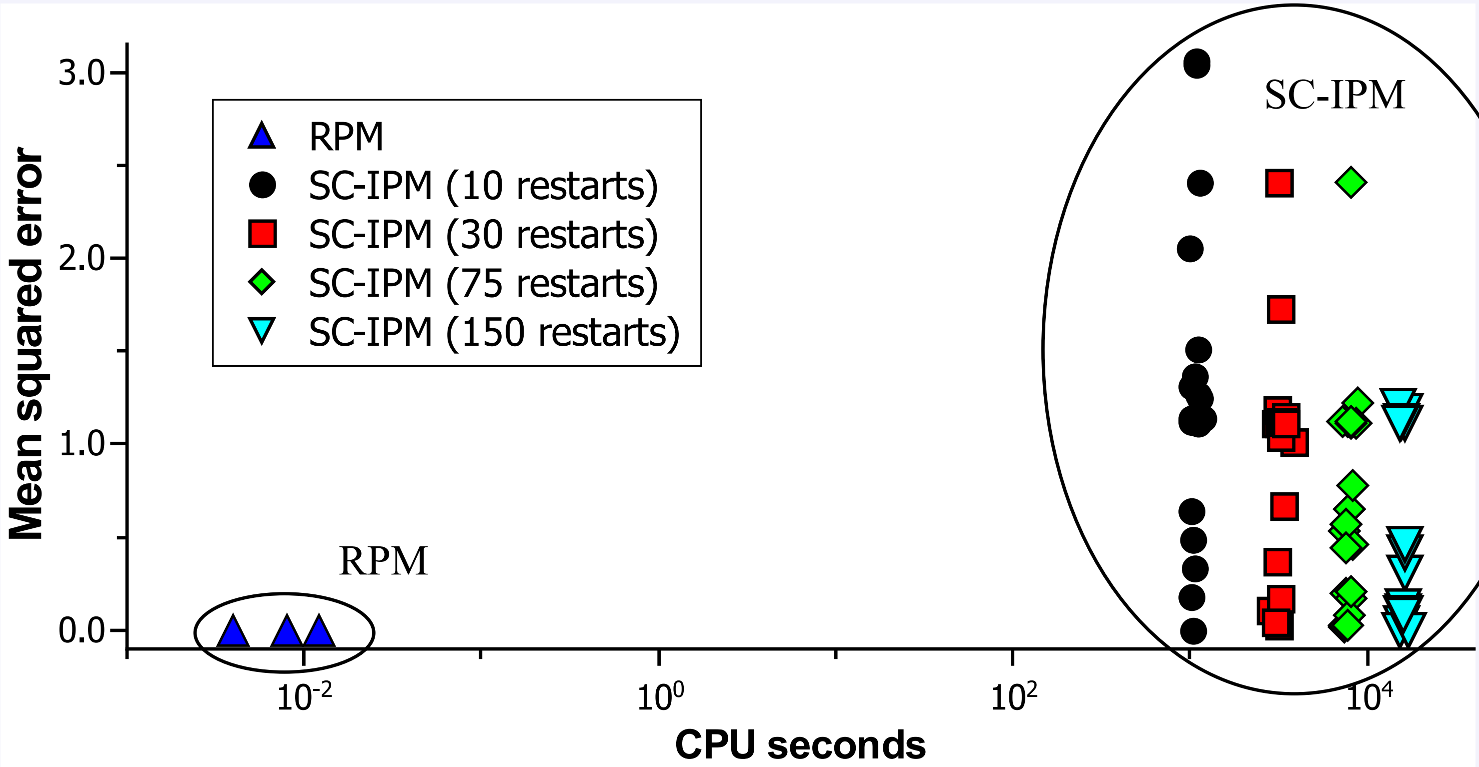
We compared RPM to SC-IPM, its predecessor, on synthetic data for a three-variable predator-prey ecosystem.



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

RPM and SC-IPM

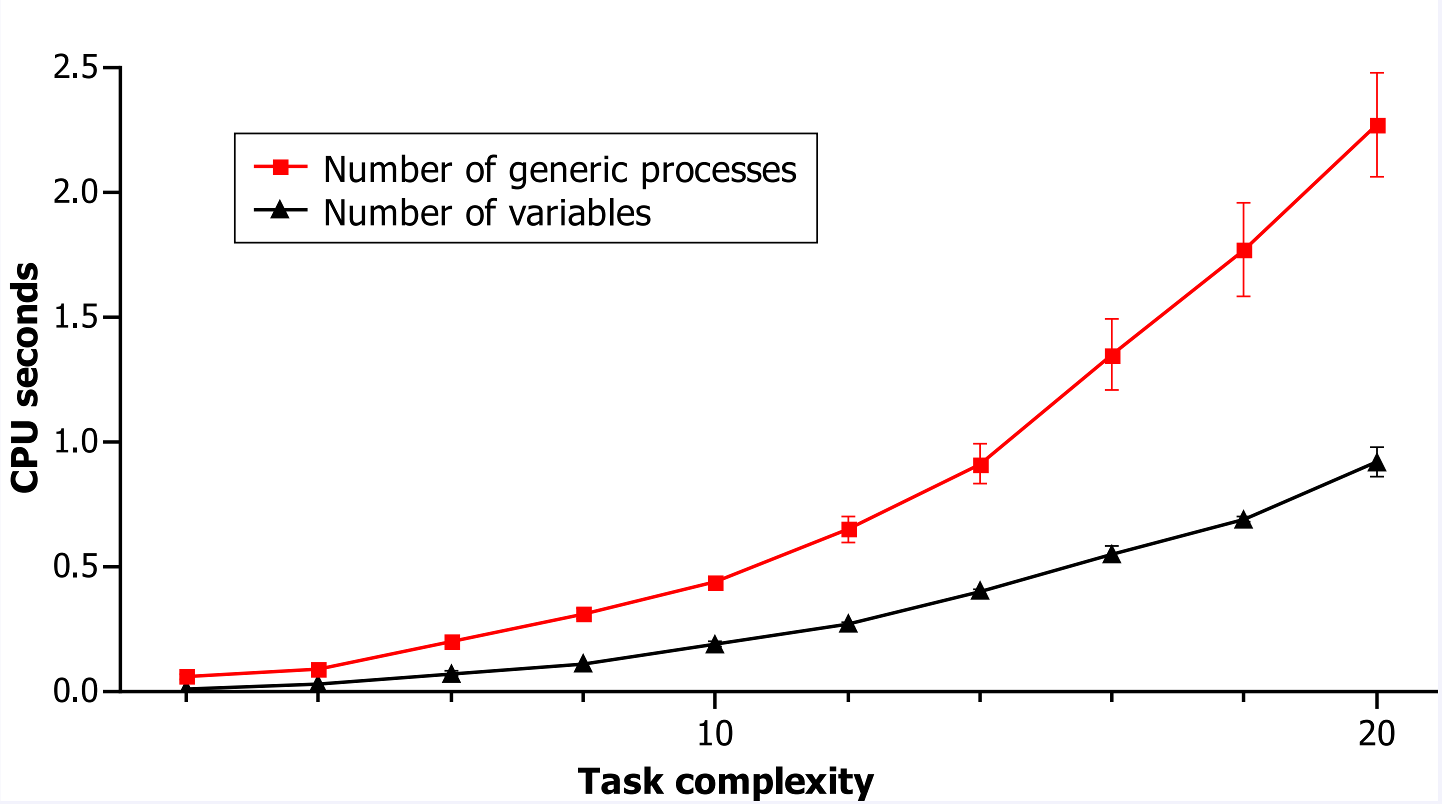
We compared RPM to SC-IPM, its predecessor, on synthetic data for a three-variable predator-prey ecosystem.



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

Handling Noise and Complexity

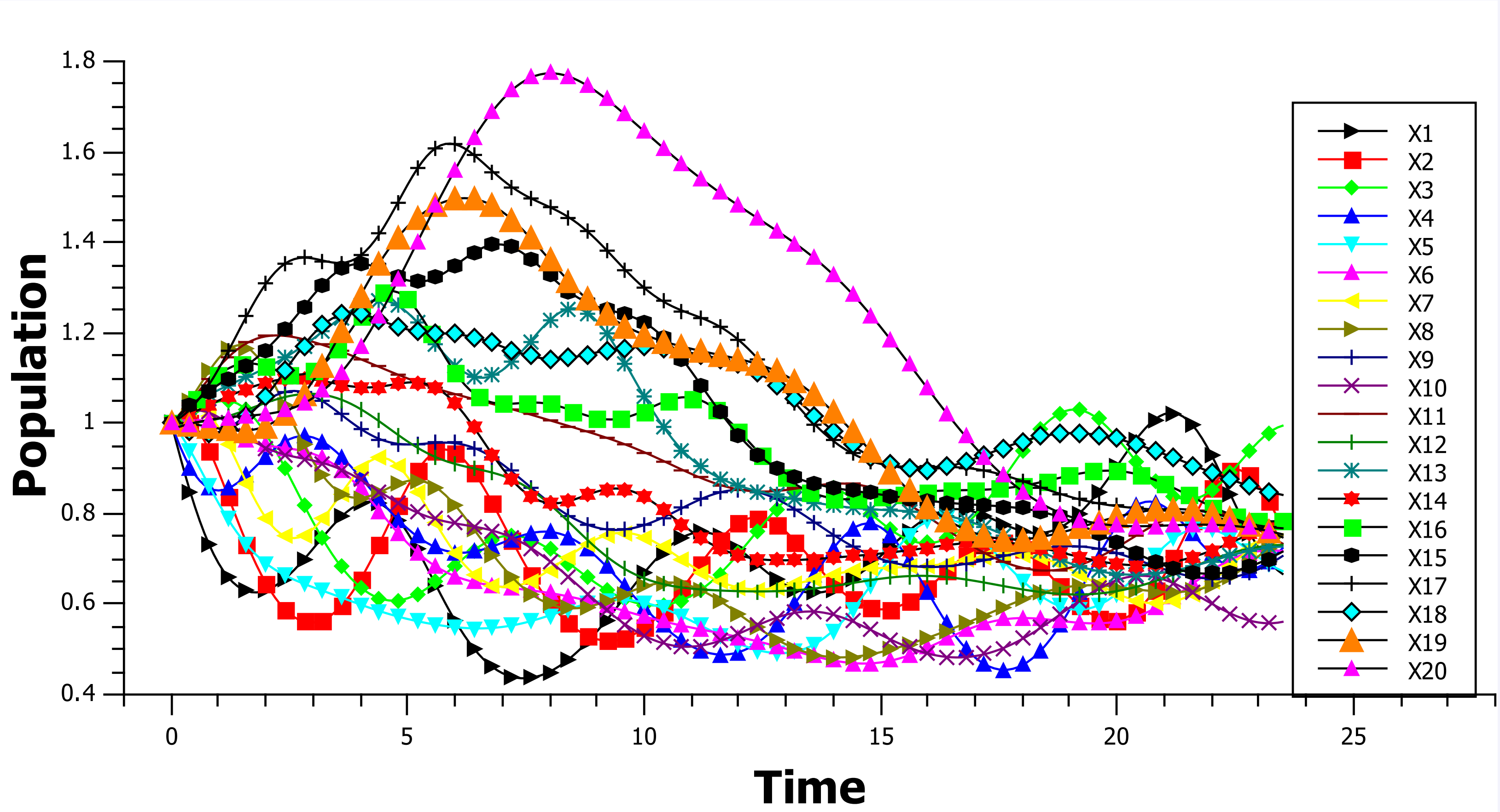
With smoothing, RPM can handle 10% noise on synthetic data.



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

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.

Further Extensions

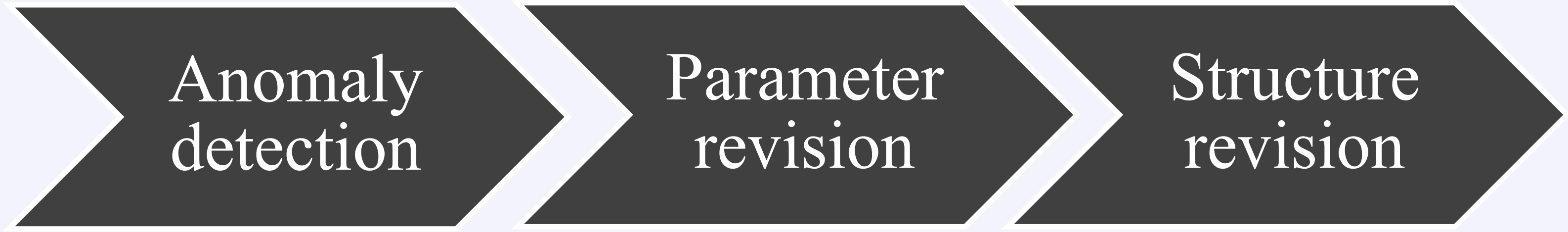
Adapting Models to New Settings

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.

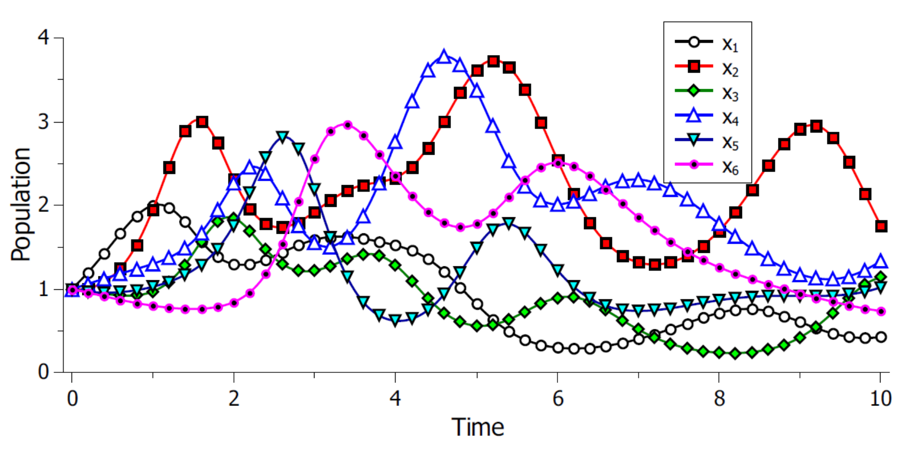


Anomaly detection

Parameter revision

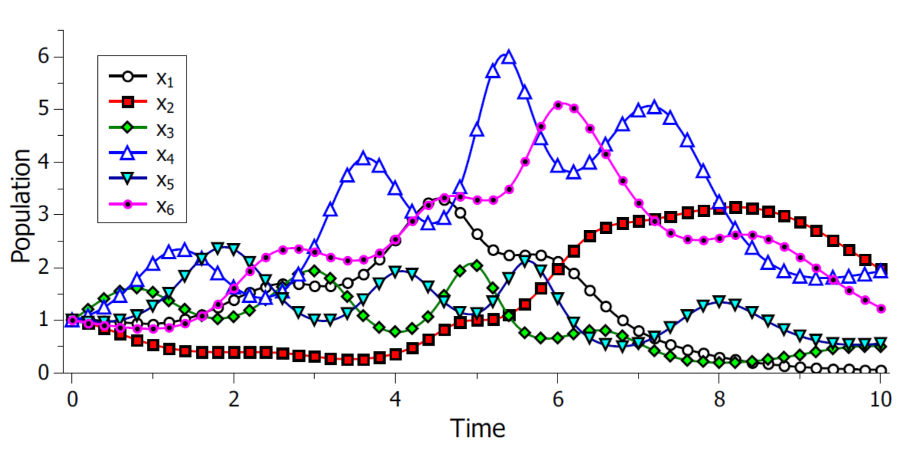
Structure revision

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



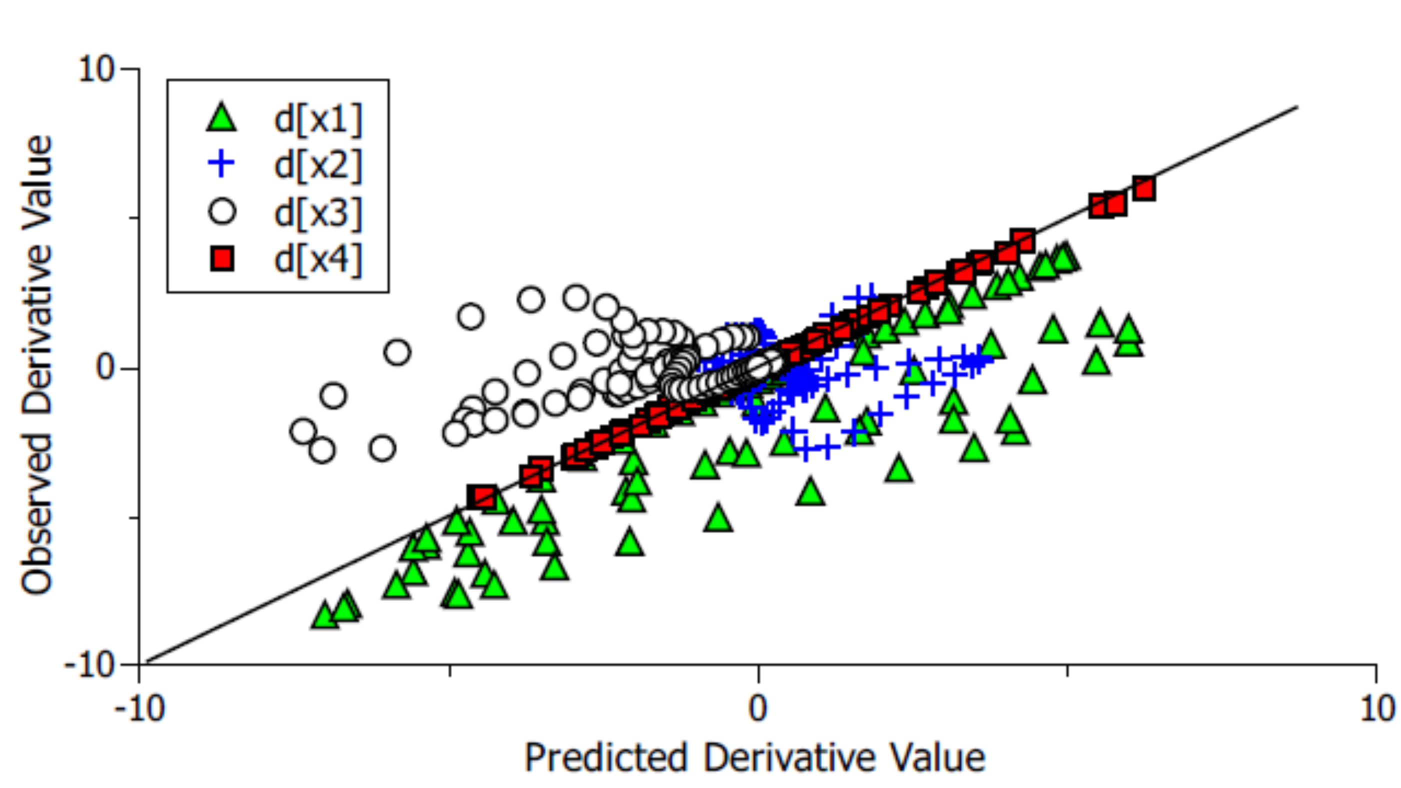
$$d[x1] = 1.7\,x1 - 0.8\,x1\,x2 - 0.8\,x1\,x3$$
$$d[x2] = 0.25\,x1\,x2 - 0.7\,x2\,x3$$
$$d[x3] = 0.8\,x2\,x3 - 0.9\,x3\,x4 + 1.1\,x1\,x3$$
$$d[x4] = 1.1\,x3\,x4 - 0.8\,x4\,x5$$
$$d[x5] = 0.8\,x4\,x5 - 1.0\,x5\,x6$$
$$d[x6] = 0.9\,x5\,x6 - 1.1\,x6$$

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.

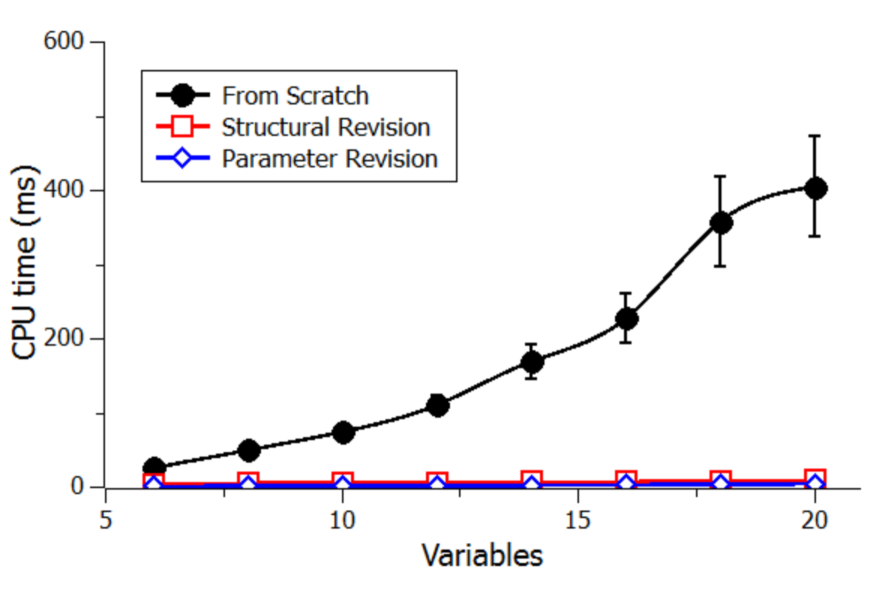
Revising a Process Model

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:

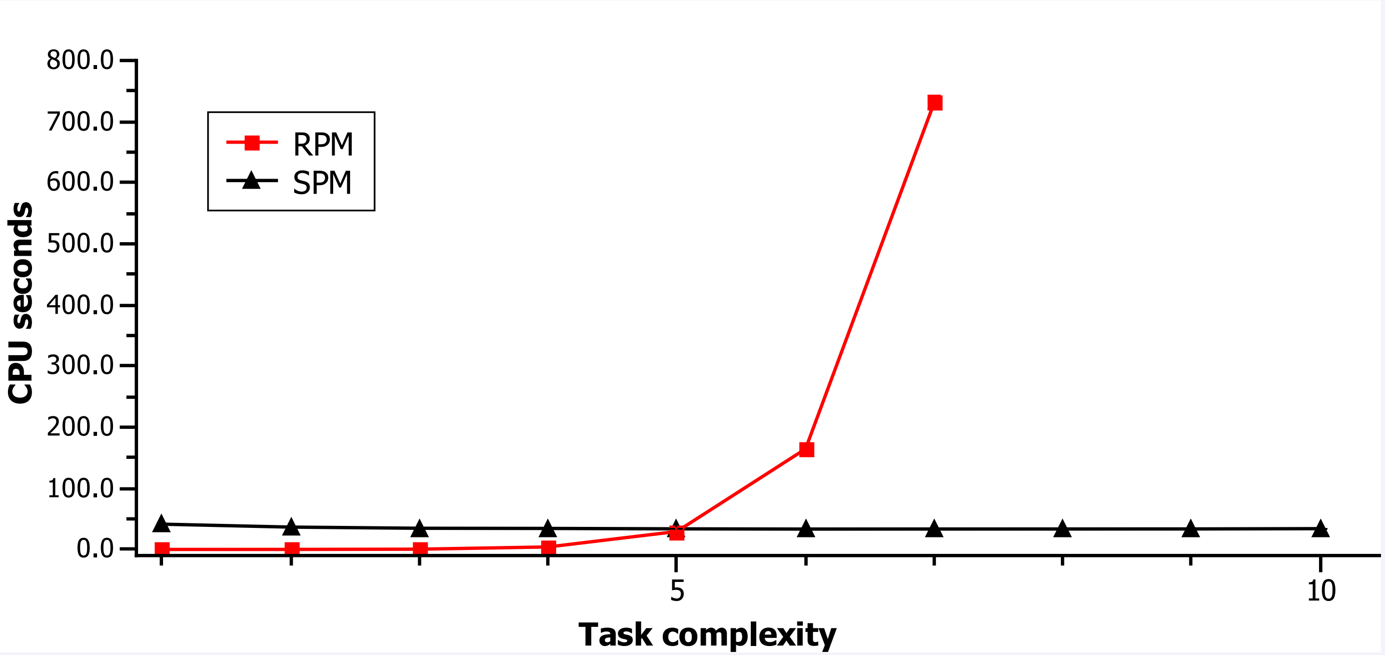
- Instantiating initially only variables in a generic process that determine its rate expression;
- Binding other variables that a process influences only when finding equations for their derivatives.

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 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 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)
- Discovering new processes (search for rate expressions)

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

Applications of Process Modeling

Scalable methods for process model induction would be useful in many practical settings, including:

- Elucidating new reaction pathways in biochemistry
- Understanding ecological dynamics of human microflora
- Designing reaction pathways for chemical production
- Designing metabolic pathways for synthetic biology

Computational tools for scientific discovery should let us not only interpret observations, but generate new behavior.

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
- Uses 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/* .

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 Inductive Process Modeling

Arvay, A., & Langley, P. (2016). Selective induction of rate-based process models. *Proceedings of the Fourth Annual Conference on Cognitive Systems*. Evanston, IL.

Arvay, A., & Langley, P. (2016). Heuristic adaptation of scientific process models. *Advances in Cognitive Systems*, *4*, 207–226.

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.

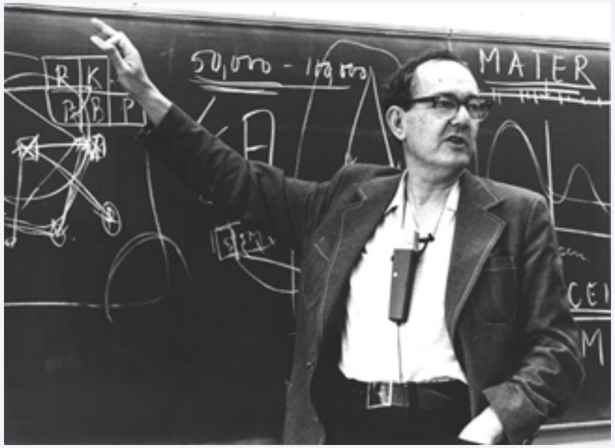
Dzeroski, S., Langley, P., & Todorovski, L. (2007). Computational discovery of scientific knowledge. In S. Dzeroski & L. Todorovski (Eds.), *Computational discovery of communicable scientific knowledge*. Berlin: Springer.

Langley, P. (2019). Scientific discovery, causal explanation, and process model induction. *Mind & Society*, *18*, 43–56.

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.

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.