Selective Induction of Rate-Based Process Models

Adam Arvay

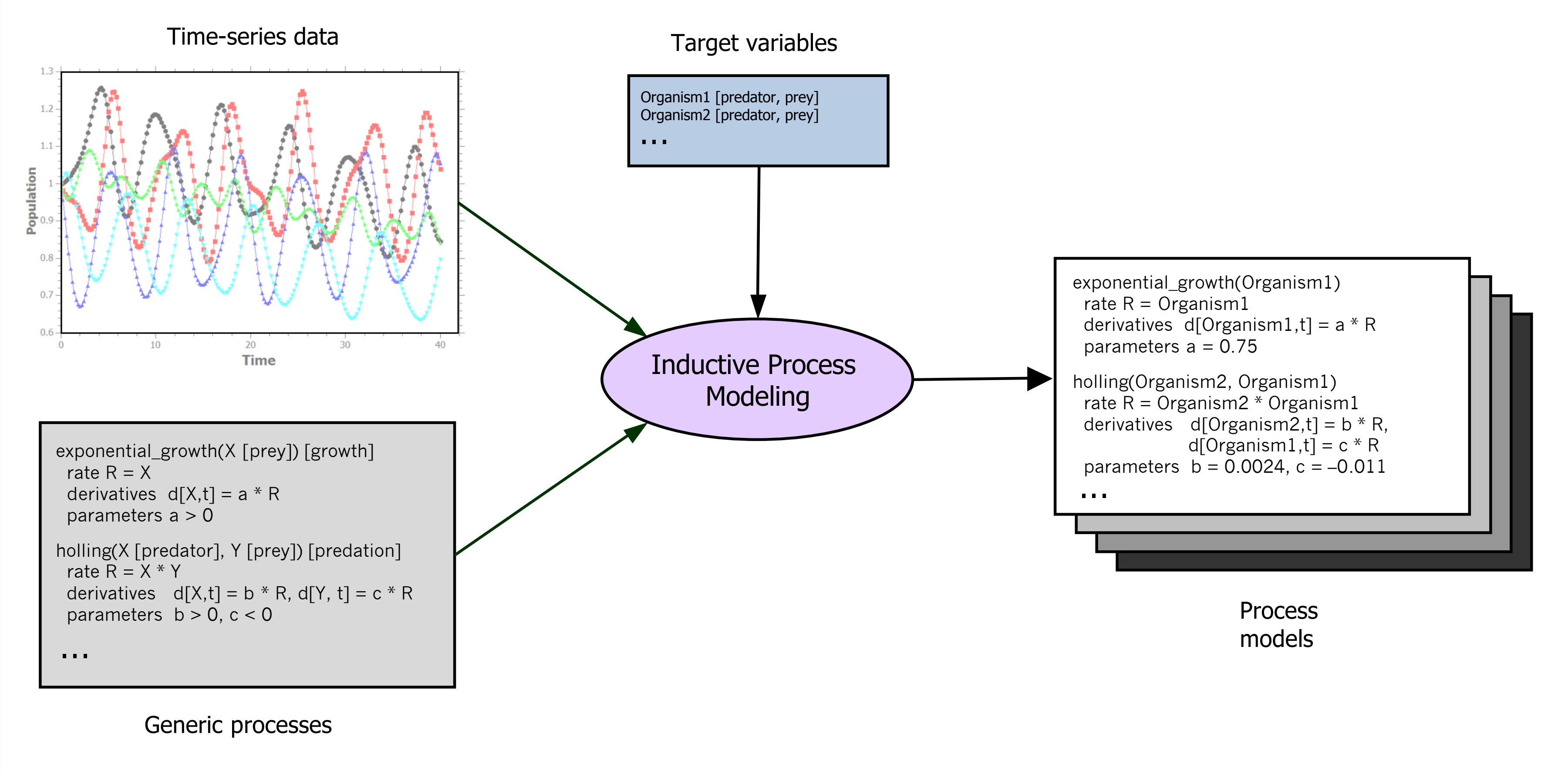
Pat Langley

Department of Computer Science
University of Auckland
Auckland, NZ

Thanks to W. Bridewell, S. Dzeroski, R. Morin, L. Todorovski, and others for their contributions to this research, which was funded by ONR Grant N00014-11-1-0107.

Inductive Process Modeling

Inductive process modeling construction of explanations for time series from background knowledge.



Time-series data

Target variables

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

Inductive Process Modeling

```
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
 …
```

Generic processes

```
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
 …
```

Process models

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

Relevance to Cognitive Systems

Research on process model induction is relevant to cognitive systems because it:

- Addresses a *high-level task* that only humans can handle;
- Uses *structured knowledge* to finds explanatory models;
- Combines abilities into an *integrated system*; and
- Utilizes *heuristic search* to make problems tractable.

These are key characteristics of cognitive systems research (Langley, 2012).

A Formalism for Process Models

A quantitative process model comprises a set of processes *P*, each of which includes:

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

This formalism has important mathematical properties that aid in model induction.

The notation borrows directly from 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
```

These units serve as *building blocks* for constructing models.

RPM: Regression-Guided Process Modeling

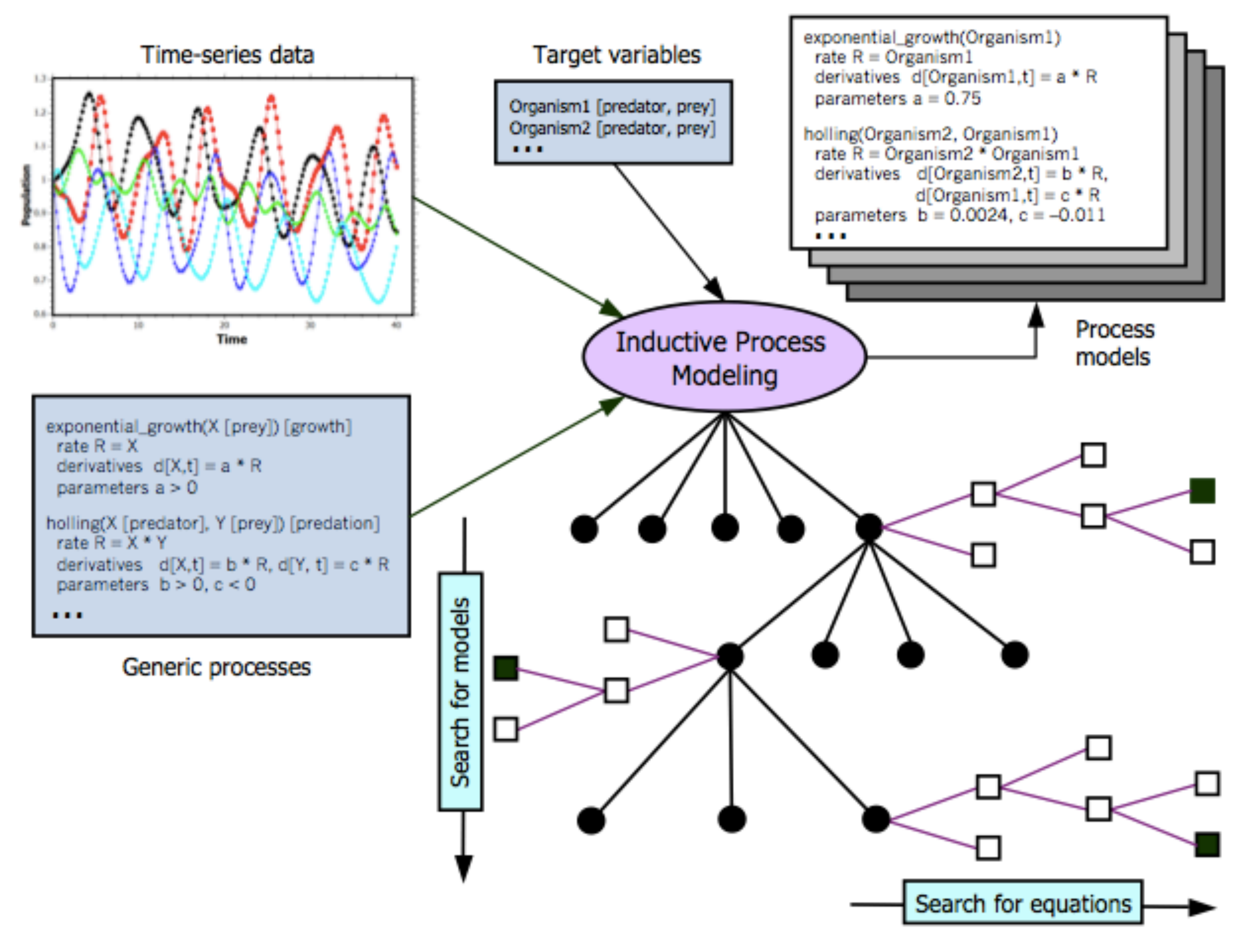
RPM (Langley & Arvay, 2015) is a system for process model induction that:

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



Time-series data

Target variables

```
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
  ...
```

Process models

Inductive Process Modeling

```
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
  ...
```

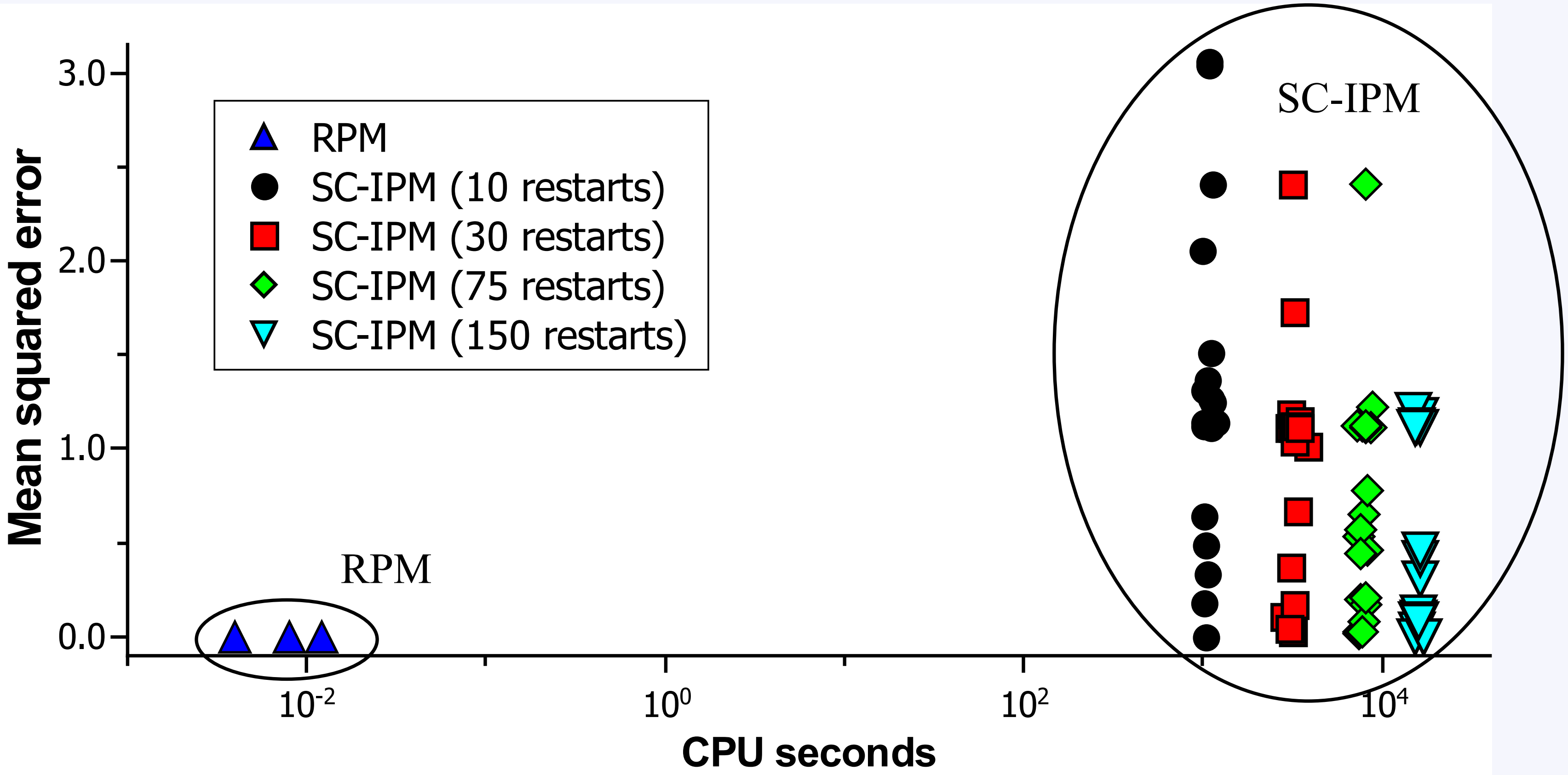
Generic processes

Search for models

Search for equations

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.

Three Drawbacks of RPM

Despites these advantages, RPM suffers from three problems:

- Generates all process instances at initialization time
  - Combinatorial number of instantiations
  - Some process instances have the same rates
- Carries out exhaustive search for differential equations
  - Practical for sparsely connected process models
  - Intractable for equations with more than five terms
- Relies on greedy search through the space of models
  - Later equations constrained by earlier ones
  - But system can still find poor sets of equations

These led us to develop SPM, an extended system for process model induction.

Selective Induction of Process Models

SPM incorporates three extensions that respond directly to the limitations of RPM:

- *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.

Delayed Variable Binding

RPM cannot induce certain 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;
  - E.g., given a process with variables A, B, C, and D with the rate expression A * B, SPM instantiates only A and B.
- Binding other variables that a process influences only when finding equations for their derivatives.

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

Increased Model Coverage

Claim: *SPM induces a superset of the models found by RPM that adequately explain the observations*.

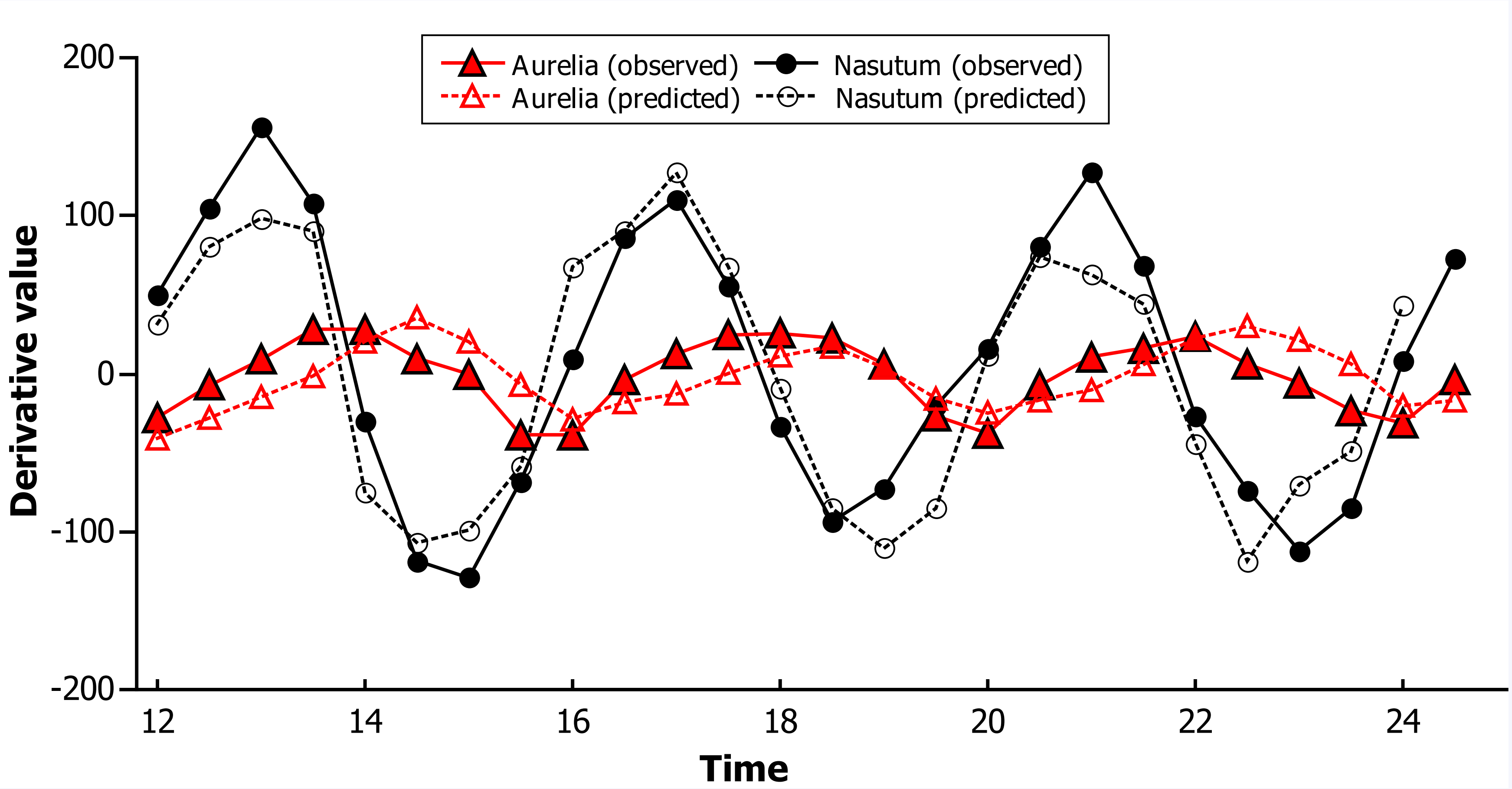
We ran RPM and SPM on five different ecological time series, both natural and synthetic.

- In all cases, both systems found models with high accuracy;
- Also, for synthetic data, they reconstructed the target model.

Thus, SPM’s more selective approach does not keep it doing well on problems that RPM can handle.

Behavior on Natural Data

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

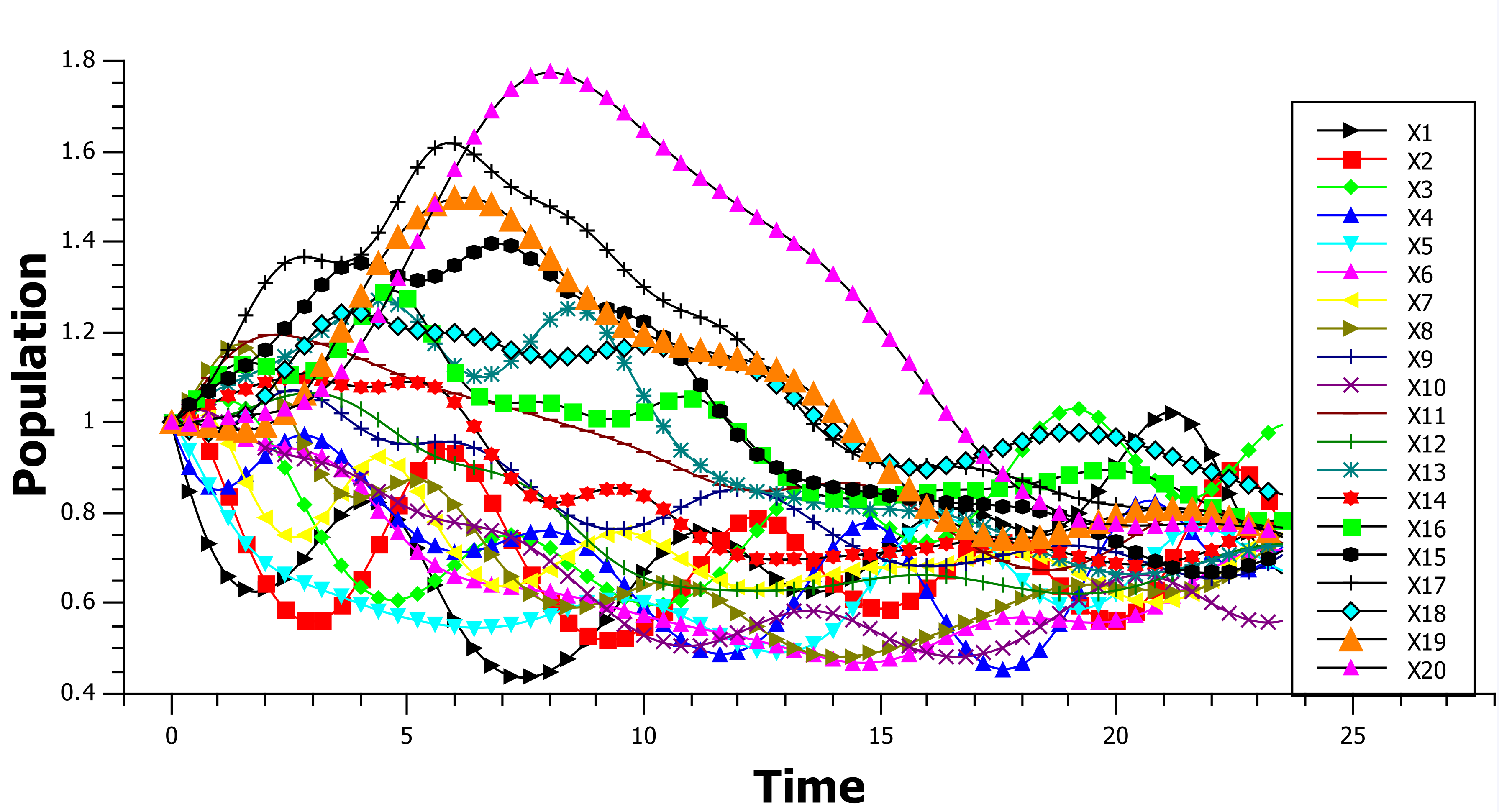


*d[aurelia] = 0.75 * aurelia − 0.11 * nasutum * aurelia [r*^{2} = 0.84]

*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.



Both systems scale well to modeling tasks with many variables.

Increased Model Coverage

Claim: *SPM induces a superset of the models found by RPM that adequately explain the observations*.

We also ran RPM and SPM on a number of synthetic data sets for chemical reaction pathways.

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

RPM could not induce any of the models, while SPM found them without difficulty.

Heuristic Search for Equations

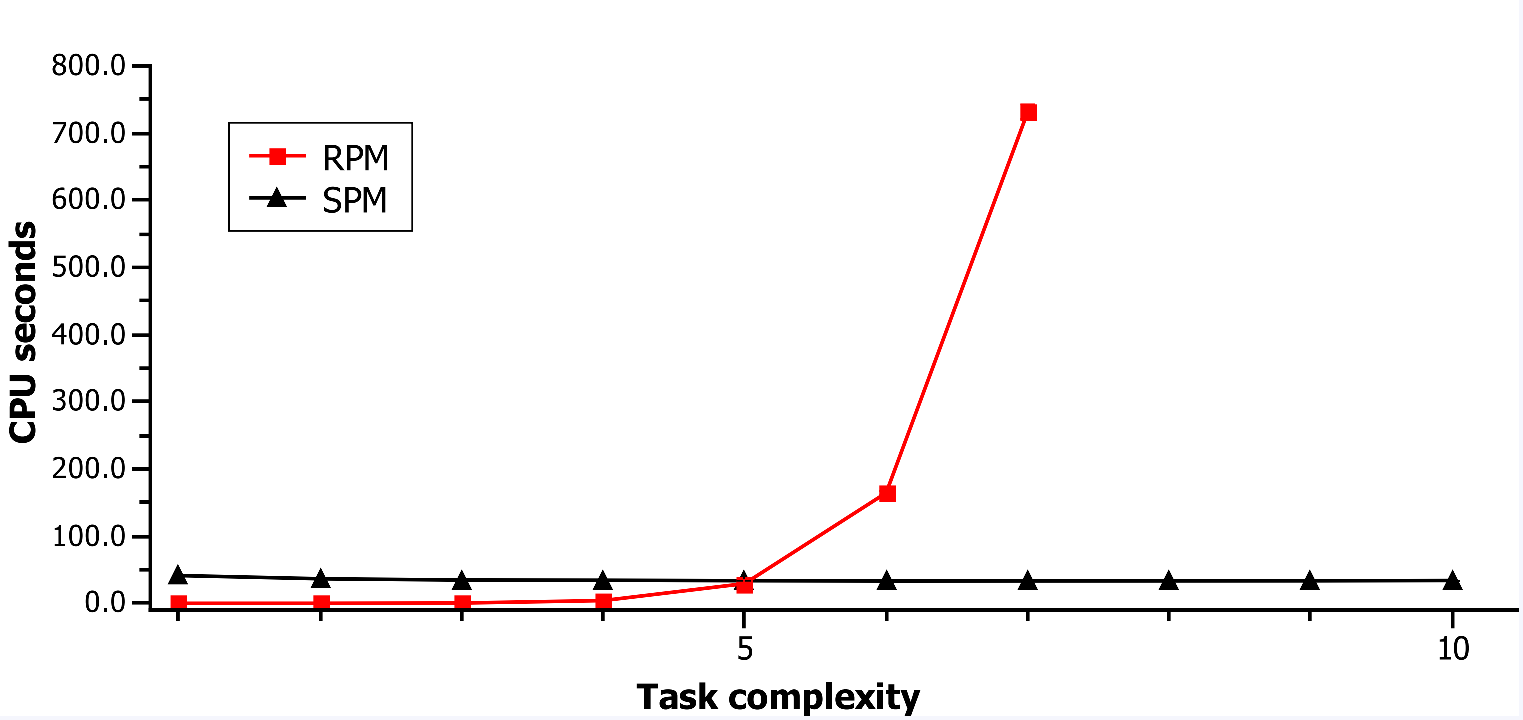
RPM’s exhaustive search for individual equations does not scale well; SPM avoids this problem by:

- *Selecting a subset* of processes (with rates) as input to multiple linear regression;
- Carrying out *backward elimination* to identify which processes to retain in the equation;
- Repeating these steps many times to increase chances of finding an equation with appropriate terms.

Sampling is necessary because the variables in our data sets are highly *collinear*, which makes coefficients inaccurate.

Better Scaling to Equation Complexity

As the number of terms in a target equation increases, induction time for SPM will grow more slowly than for RPM.



RPM’s exhaustive search rapidly becomes intractable; SPM’s approach runs in time linear with equation complexity.

Search for Consistent Process Models

RPM’s greedy search sometimes leads it down dead ends, so it fails to find accurate models.

SPM avoids this problem by organizing its search differently:

- Finding multiple differential equations for each target variable;
- Considering all ways to combine them into consistent models that satisfy process constraints.

This strategy should increase SPM’s probability of inducing one or more accurate models.

Increased Reliability

Claim: *SPM induces a more complete set of process models than RPM and has greater chances of recovering the target*.

	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

SPM’s strategy increased its probability of inducing models of chemical reaction pathways.

The system also found multiple models with similar accuracies.

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

We have reported an approach to inductive process modeling that extends earlier work by:

- Delaying binding of variables in generic processes
- Carrying out heuristic search for component equations
- Utilizing more extensive search for consistent models

We also described a new system, SPM, that incorporates these ideas and demonstrated its benefits experimentally.

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

End of Presentation