Using Domain Specific Knowledge for Automated Modeling

Ljupčo Todorovski and Sašo Džeroski

Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
{Ljupco.Todorovski,Saso.Dzeroski}@ijs.si

Abstract. In this paper, we present a knowledge based approach to automated modeling of dynamic systems based on equation discovery. The approach allows for integration of domain specific modeling knowledge in the process of modeling. A formalism for encoding knowledge is proposed that is accessible to mathematical modelers from the domain of use. Given a specification of an observed system, the encoded knowledge can be easily transformed into an operational form of grammar that specifies the space of candidate models of the observed system. Then, equation discovery method LAGRAMGE is used to search the space of candidate models and find the one that fits measured data best. The use of the automated modeling framework is illustrated on the example domain of population dynamics. The performance and robustness of the framework is evaluated on the task of reconstructing known models of a simple aquatic ecosystem.

1 Introduction

Scientists and engineers build mathematical models in order to analyze and better understand the behavior of real world systems. Mathematical models have the ability to integrate potentially vast collections of observations into a single entity. They can be used for simulation and prediction of the behavior of the system under varying conditions. Another important aspect of mathematical models is their ability to reveal the basic processes that govern the behavior of the observed system.

Establishing an acceptable model of an observed system is a very difficult task that occupies a major portion of the work of the mathematical modeler. It involves observations and measurements of the system behavior under various conditions, selecting a set of system variables that are important for modeling, and formulating the model itself. This paper deals with automated modeling task, i.e., the task of formulating a model on the basis of observed behavior of the selected system variables. A framework for automated modeling of real world systems is based on equation discovery methods [6, 10, 12].

Most of the work in equation discovery is concerned with assisting the empirical (data driven) approach to modeling physical systems. Following this approach, the observed system is modeled on a trial-and-error basis to fit observed data. The scientist first chooses a structure of the model equations from some general class of structures (such as linear or polynomial) that is believed to be adequate, fits the constant parameters, and checks whether the simulation match the observed data. If not, the procedure is repeated until an adequate model is found. None of the available domain knowledge about the observed system (or a very limited portion thereof) is used in the modeling process. The empirical approach is in contrast to the theoretical (knowledge driven) approach to modeling, where the basic physical processes involved in the observed system are first identified. A human expert then uses domain knowledge about the identified processes to write down a proper structure of the model equations. The values of the constant parameters of these equations are fitted against the observed data using standard system identification methods [7]. If the fit of the model to the observed data is not adequate, human expert can decide to go through a feedback loop and refine the proposed structure of the model equations.

While most of the equation discovery methods are data driven, here we present a knowledge driven method capable of integrating domain specific modeling knowledge in the process of equation discovery. The presented method allows for integration of theoretical and empirical approaches to automated modeling of real world systems. To achieve that goal, we first present a formalism for encoding modeling knowledge from a domain of interest in a form of taxonomy of process classes. Given a specific modeling task, the encoded knowledge can be easily transformed in an operational form of grammar that specifies the space of candidate models for the given task. This grammar representation allows us to use the existing equation discovery method Lagramge [10] in order to search through the space of candidate models and find the one that fits the measurement data best.

The paper is organized as follows. The formalism for encoding domain specific modeling knowledge is presented in Section 2. The automated modeling framework based on the equation discovery method Lagramge is presented in Section 3. The results of experiments on several tasks of reconstructing several models of a simple aquatic ecosystem are presented in Section 4. Finally, Section 5 concludes the paper with a summary, discusses the related work and proposes directions for further work.

2 Encoding Domain Specific Modeling Knowledge

The formalism for encoding domain specific modeling knowledge allows for organization of the knowledge in three parts. The first part contains knowledge about what types of variables are measured and observed in systems from the domain of interest. The second part contains models of typical basic processes that govern the behavior of systems in the observed domain. The third part encodes knowledge about how to combine models of individual basic processes into a single model of the whole system.

The use of the formalism will be illustrated on population dynamics domain. Note that focusing on a single domain is due to the lack of space and does not correspond to the generality of the formalism. The presented formalism allows for encoding modeling knowledge from variety of domains, including chemical kinetics and spring mechanics. The formalism is also general enough to allow representing domain independent knowledge about measurement scales used to measure the system variables. The use of the formalism in these domains is further elaborated in [9].

Population ecology studies the structure and dynamics of populations, where each population is a group of individuals of the same species sharing a common environment. In this paper, we consider modeling the dynamics of populations, especially the dynamics of change of their concentrations (or densities) in the observed environment. The models take the form of systems of differential equations [8].

2.1 Taxonomy of variable types

The first part of the domain specific modeling knowledge is the taxonomy of types of the variables that can be used in the models. An example of a taxonomy of types for the population dynamics domain is presented in Table 1.

Table 1. A taxonomy of variable types that are can be used in population dynamics models.

```
type Concentration is nonnegative_real_number
type Inorganic is Concentration
type Population is Concentration
```

The generic variable type in the population dynamics domain is `Concentration`, since we are interested in the change of concentrations of different populations. The definition of the concentration type specifies that concentrations are nonnegative real numbers. The concentration type has two sub-types. The first is `Population`, used to denote a concentration of organic species. The second type `Inorganic` is used to denote a concentration of an inorganic nutrient that can be consumed by organic species. Note that the inheritance rule applies to the taxonomy of types, resulting in the fact that the population and inorganic types are also nonnegative real numbers.

2.2 Taxonomy of basic process classes

The most important part of the modeling knowledge is the taxonomy of process classes. Each of them represents an important class of basic processes that govern or influence the behavior of dynamic systems in the domain of interest. The complete taxonomy of process classes in the population dynamics domain is presented in Figure 1.

On the highest level the taxonomy distinguishes between processes that involve single species or inorganic nutrient and processes that represent interactions between two or more species and nutrients. Down the taxonomy tree, the process classes become more specific. Each leaf node represents a particular modeling alternative usually used by the experts in the domain. The formalization of several process classes from the taxonomy is presented in Table 2.

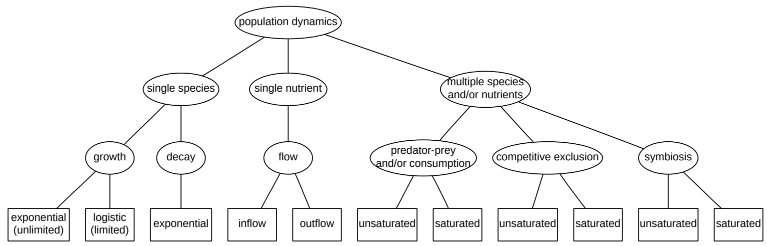

Fig. 1. A taxonomy of classes of processes used for modeling population dynamics systems.

Table 2. A formalization of several process classes from the population dynamics taxonomy presented in Figure 1.

```
process class Growth(Population p)
process class Exponential_growth is Population_growth
  expression const(growth_rate,0,1,Inf) * p
process class Logistic_growth is Population_growth
  expression const(growth_rate,0,1,Inf) * p *
    (1 - p / const(capacity,0,1,Inf))

process class Feeds_on(Population p, set of Concentration cs)
  condition p ∉ cs
  expression p * ∏_{c∈cs} Saturation(c)
```

The first process class `Growth` represents the processes of a single species growth in the absence of any interaction with other populations in the observed environment. The `Growth` process class has two sub-classes, specifying two modeling alternatives for the growth processes. The `Exponential_growth` class specifies unlimited exponential growth of the observed population. For many real-world systems the unlimited growth model is appropriate, since real-world environments typically have limited carrying capacity for the population. In such cases, alternative `Logistic_growth` model is used.

The definition of each process class consist of three parts, each of them specifying one aspect of the processes in the class and/or the models thereof. Each part is presented in more detail below.

Types of the variables involved. This part of the process class definition is used to specify what are the types of the variables that are involved in the processes from the class. For example, each process from the `Growth` process class involves a single population `p`. Furthermore, the processes in the `Feeds_on` class involve a population `p` that depends on several food sources specified as a set of concentrations `cs`. The declarations of the types of variables are inherited through the taxonomy, i.e., processes in the `Exponential_growth` class inherit from the parent class `Growth` the fact that they involve a single variable `p` of `Population` type.

Conditions on the variables involved. This part of the process class definition specifies constraints on the variables involved in the processes. The condition `p` $\notin$ `cs` in the `Feeds_on` process class forbids cannibalism within a single species, i.e., specifies that the population can not predate on itself.

Declaration of the process models. This part of the process class definition specifies the equation template that is used by domain experts for modeling processes in the process class. The equation template can include variables involved in the process, as well as constant parameters. The values of the constant parameters can be fitted against the measurements of the observed system. In the equation template, symbol `const(name, lower_bound, default, upper_bound)` is used to specify a constant parameter. The symbol specifies the name of the constant parameter, along with the lower and upper bound on its value, as well as its initial or default value.

Table 3. A taxonomy of function classes specifying different expressions for modeling the saturation of predation rate in the population dynamics domain.

```
function class Saturation(Concentration c)
function class No_saturation is Saturation
  expression c
function class Saturation_type_1 is Saturation
  expression c / (c + const(saturation_rate(0,1,Inf)))
```

Note that the expression used to declare the model of the `Feeds_on` processes involves `Saturation(c)` expression. It is used to denote whether the predation (or grazing) rate is saturated or not. The latter means that the predation is not unlimited, i.e., when the prey (or nutrient) population becomes abundant, the predation capacity saturates to some limit value. The last function class in Table 3 specifies one of different alternative expressions used for modeling the saturation [8]. The second function class specifies an expression used for modeling unsaturated (unlimited) predation. The declarations of functions have the same structure as declarations of process classes. The difference is that function classes are not used to define processes, but rather specify expression templates that are commonly used in the model templates.

2.3 Scheme for combining models of individual processes

The modeling knowledge formalism also specifies the scheme that is used to combine the models of individual processes into a model of the whole system. Two combining schemes used to combine models of basic processes in the population dynamics domain are presented in Table 4.

The first combining scheme specifies how the equation that models the time change of a concentration of an inorganic nutrient `i` is composed from the individual process models. The change of `i` (assessed by the time derivative $d\mathtt{i}/d\mathtt{t}$ of `i`) decreases with all the `Feeds_on` interactions, in which a population `p` consumes the inorganic nutrient `i`.

The second scheme is a bit more complicated. The change of the concentration of the population `p` (again assessed by the time derivative of `p`) increases

Table 4. Combining schemes that specify how to combine the models of individual population dynamics processes into a model of the whole system.

`combining scheme Population_dynamics(Inorganic i)`
$d\mathtt{i}/d\mathtt{t} = -\sum_{\mathtt{food},\mathtt{i}\in\mathtt{food}}$ `const(_,0,1,Inf) * Feeds_on(p, food)`
`combining scheme Population_dynamics(Population p)`
$d\mathtt{p}/d\mathtt{t} = +$`Growth(p)` $-$ `Decay(p)`
$+\sum_{\mathtt{food}}$ `const(_,0,1,Inf) * Feeds_on(p, food)`
$-\sum_{\mathtt{pred},\mathtt{food},\mathtt{p}\in\mathtt{food}}$ `const(_,0,1,Inf) * Feeds_on(pred, food)`

with the population growth and decreases with the decay. On the other hand, Feeds_on processes can positively or negatively influence the change, depending on the role of p in the interaction. The processes where p is involved as a consumer or predator, positively influence the change of p, while the processes where p is involved as a prey (or nutrient), negatively influence the change of p (see the last two lines in Table 4).

3 An automated modeling framework based on equation discovery

The integration of domain specific modeling knowledge in the process of equation discovery is schematized in Figure 2. The integration is supported using the transformation principle. First, the domain specific knowledge is transformed into a grammar based on modeling task specification. The obtained grammar specifies the space of candidate models for the observed system. The equation discovery method LAGRAMGE [10] can be then used to search through the space of candidate models and find the one that fits the measured data best.

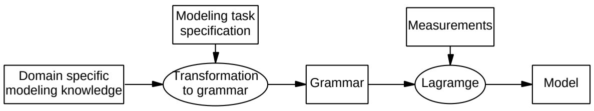

Fig. 2. An automated modeling framework based on the integration of domain specific modeling knowledge in the process of equation discovery.

3.1 Modeling task specification

The encoded modeling knowledge is general, in the sense that it allows for modeling of an arbitrary system from the population dynamics domain. In order to use the knowledge for automated modeling of a particular system, a specification of the system has to be provided first. An example of modeling task specification is presented in Table 5.

A specification consists of two parts. The first part specifies the list of observed system variables. Each variable is assigned a type from the taxonomy of types. The example of a simple aquatic ecosystem involves inorganic nutrient,

Table 5. A task specification used for modeling a simple aquatic ecosystem consisting of two consumption interactions between three populations of inorganic nutrient, phytoplankton, and zooplankton.

```
variable Inorganic nut
variables Population phyto, zoo

processes Decay(phyto), Decay(zoo)
processes Feeds_on(phyto, nut), Feeds_on(zoo, phyto)
```

phytoplankton and zooplankton. The second part of the modeling task specification consist of a list of processes that govern the dynamics of the observed system. Each process in the list should belong to a process class from the taxonomy of process classes: the process class and the system variables involved in the process are specified. The two `Decay` processes specify that the populations of phytoplankton and zooplankton tend to decrease in absence of any interactions with the environment and other species. The two `Feeds_on` processes specify that phytoplankton consumes inorganic nutrient and zooplankton consumes phytoplankton.

3.2 Transforming the modeling knowledge into a grammar

The algorithm for transforming the encoded modeling knowledge into grammar takes two arguments as input. The first is the library of modeling knowledge for the domain at hand, and the second is the modeling task specification, specifying the types of the variables and the processes of the observed system.

The transformation of the modeling knowledge proceeds in a top-down manner. It starts with the starting symbol and assigns productions to it, then proceeds with other nonterminal symbols. The starting symbol of the grammar corresponds to the combining schemes in the encoded modeling knowledge. It is used to combine the models of individual processes into a single model of the whole observed system. The other nonterminal symbols in the grammar correspond to process classes. Alternative productions for each nonterminal symbol specify alternative models of the corresponding process class.

For example, consider the aquatic ecosystem example from Table 5. The start symbol will use the combining schemes from Table 4 to compose a model of the whole ecosystem as illustrated in Table 6. The right hand side of the production for the starting symbol builds three equations, one for each variable of the observed ecosystem. Each equation reflects the corresponding combining scheme from Table 4. For example, the second equation is built by summing up the growth processes for the `phyto` population (the leading `0` means that no such process is specified), the `phyto` decay processes (i.e., `Decay(phyto)`, the consumption processes where `phyto` is involved as consumer (i.e., `Feeds_on(phyto, {nut})`), and the predator-prey processes where `phyto` has the role of prey (i.e., `Feeds_on(zoo, {phyto})`). The productions for the other nonterminal symbols are built according to the taxonomies of process and function classes from Tables 2 and 3.

Table 6. A part of the grammar specifying the candidate models for modeling the simple aquatic ecosystem presented in Table 5.

```
Start ->
  time_deriv(nut) = - const[_:0:1:] * Feeds_on_phyto_nut;
  time_deriv(phyto) = 0 - Decay_phyto + const[_:0:1:] * Feeds_on_phyto_nut
    - const[_:0:1:] * Feeds_on_zoo_phyto;
  time_deriv(zoo) = 0 - Decay_zoo + const[_:0:1:] * Feeds_on_zoo_phyto

Feeds_on_phyto_nut -> phyto * Saturation_nut;
Feeds_on_zoo_phyto -> zoo * Saturation_phyto;

Decay_phyto -> Exponential_decay_phyto;
Exponential_decay_phyto -> const[decay_rate:0:1:] * phyto;

Saturation_nut -> No_saturation_nut | Saturation_type_1_nut;
No_saturation_nut -> nut;
Saturation_type_1_nut -> nut / (nut + const[saturation_rate:0:1:]);
```

The grammar from Table 6 specifies 16 possible model structures of the observed system. In the final step from Figure 2, Lagramge is used to fit the parameters of each model structure against measured data and find the model that fits the data best. Strictly speaking, the grammar in Table 6 is not context free. The production for the starting symbol generates two `Feeds_on_phyto_nut` symbols, one in the first and another one in the second equation. In a context free grammar, these two nonterminal symbols can generate two different expressions. In population dynamics models, however, these two expressions have to be the same. Adding context dependent constraints to the grammar can overcome this limitation of context free grammars used in previous versions of Lagramge [10].

4 Experimental evaluation

The experiments are performed that evaluate the performance of the proposed automated modeling framework on 16 different tasks of reconstructing 16 models of the simple aquatic ecosystem, specified in Table 5, from simulated data. This experimental setup is useful since it allows a comparison of the discovered model with the original one and an evaluation of the success of reconstruction. This is the first evaluation criterion used in the experiments.

The synthetic experimental data is obtained by simulating the 16 chosen models. Each of the models is simulated from ten randomly chosen initial states to obtain ten simulation traces.[1] These traces are then used for a cross-validation estimate of the model error on test data, unseen during the discovery process. Root mean squared error (RMSE) was used to measure the quality of the reconstructed models. The RMSE measures the discrepancy between the provided

[1] The initial values of the system variables were randomly chosen using a random variable uniformly distributed over the $[0.5, 1.5]$ interval. Each simulation trace included 100 time steps of 0.1 time units.

values of the system variables and the values obtained by simulating the discovered model. The cross-validated RMSE of the model is the second evaluation criterion used in the experiments.

In addition to reconstructing models from noise free simulation data, we also tested the noise robustness of our automated modeling approach. To each simulation trace, artificially generated random Gaussian noise is added. The noise at five relative noise levels is added: 1%, 2%, 5%, 10%, and 20%. The relative noise level of $l\%$ means that we replaced the original value x with the noisy value of $x \cdot (1 + l \cdot r/100)$, where r is a normally distributed random variable with mean zero and standard deviation one.

We compare the performance of our automated modeling framework with the performance of previous version of equation discovery method LAGRAMGE that use context free grammars and can not use the lower and upper bounds on the values of the constant parameters [2]. The results of the comparison are summarized in Figure 3 and Table 7.

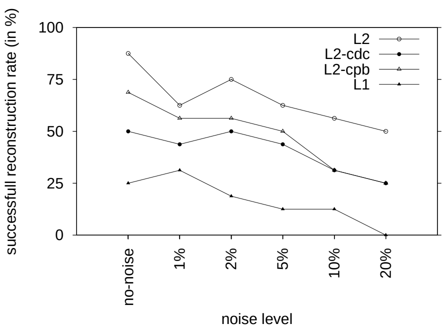

Fig. 3. The percentage of successfully reconstructed model structures (out of 16) with different versions of the grammar. Legend: `L2` = using completely constrained context dependent grammar, `L2-cdc` = using context free grammar, `L2-cpb` = using context dependent grammar without bounds on the values of the constant parameters, and `L1` = using completely unconstrained grammar (as in previous version of LAGRAMGE).

The graph in Figure 3 shows that the use of both context dependent grammars and constraints (bounds) on the values of the constant parameters contribute to the noise robustness of our automated modeling approach. The `L2` line in the graph shows that LAGRAMGE with completely constrained grammar was able to reconstruct almost 90% of the original models (14 out of 16) from noise free data and 50% of the original models from noisy data with 20% relative noise. The comparison of `L2` line with each of the `L2-cdc` and `L2-cpb` lines shows that the use of both kinds of constraints in the grammar are almost equally important. When context free grammar is used (`L2-cdc`), only 50% of the original models are successfully reconstructed from noise free data. Note that this reconstruction rate is equal to the one gained using completely constrained grammar on the data with 20% relative noise. Similarly, the successful reconstruction rate significantly drops if no bounds on the values of the constant parameters are specified (line `L2-cpb`). Note also that none of the 16 models can be success-

Table 7. The average of root squared mean errors (RMSE) of the 16 models, discovered using different constraints in the grammar from noise free data and data with added noise at five different noise levels. The RMSE of each model was estimated using 10-fold cross validation. For legend, see Figure 3.

noise level (in %)	L2	L2-cdc	L2-cpb		L1
no-noise	0.00643	0.00364	0.00404		0.00379
1	0.00880	0.00756	0.00775		0.0119
2	0.0140	0.0131	0.0132		0.0131
5	0.0319	0.0310	0.0310		0.0309
10	0.0293	0.0278	0.0279	(*1)	0.118
20	0.0549	0.0629	0.0458	(*5)	0.0647

fully reconstructed from the data with 20% relative noise using the completely unconstrained grammar (`L1`).

Furthermore, the comparison of all four lines on the graph in Figure 3 shows that the improvements due to each of the two different kinds of constraints almost sum up. That means that each kind of grammar constraints improves different aspect of reconstruction capabilities of LAGRAMGE. Note that context dependent grammars also reduce the complexity of the space of possible models, sought by LAGRAMGE 2.0. While the context dependent grammar specifies 16 distinct model structures, the context free grammar (i.e., the one without context dependent constraints) specifies the space of 256 distinct model structures. Thus, the use of context dependent constraints reduces the complexity of the search space by factor of sixteen.

The figures in Table 7 show the average RMSE of of the 16 rediscovered models using different grammars. Note that the results are unexpected. Note that the models discovered with unconstrained versions of the grammar (i.e., `L2-cdc`, `L2-cpb`, and `L1`) are on average more accurate than the models discovered with completely constrained grammar. Why? Context free grammars allow for models with inappropriate structure where different expressions are used to model the same process in different equations. Grammars without bounds on the values of the constant parameters allow for models with inappropriate (unrealistic) values of the constant parameters (e.g., negative growth or saturation rate). By analyzing the reconstructed models, we can see that most of these inappropriate models can better fit the data than the original models with proper structure and parameter values. Thus, using RMSE only is not an appropriate criterion for validating the quality of the reconstructed models.

Finally, note that inappropriate structure or inappropriate values of the constant parameters can lead to problematic models that can not be simulated.[2] The (*N) symbols beside the last two figures from Table 7 denote that N out of 16 discovered models could not be simulated properly. On the other hand, all the models discovered using the fully constrained grammar can be properly simulated.

[2] In population dynamics domain the usual reason for inability to simulate a model is "division by zero" singularities encountered due to a negative value of the saturation rate constant parameter.

5 Summary and discussion

In this paper, we presented a knowledge based approach to automated modeling of dynamic systems based on equation discovery. The approach allows for representation and integration of domain specific modeling knowledge in the process of equation discovery. The preliminary experimental evaluation shows the importance of the modeling knowledge for the performance of the approach. Using domain knowledge to constrain the space of possible model structures and possible values of the constant parameters can greatly improve the noise robustness of the equation discovery methods. Due to the lack of space, the use of the framework was illustrated on a single experimental series in the population dynamics domain only. Further experiments show that the approach is capable of reconstructing models of more complex aquatic systems including different kinds of interactions between species [9]. Note however, that the presented encoding formalism is general enough to allow for encoding and integration of different kinds of modeling knowledge from a variety of domains [9].

The presented approach follows the compositional modeling (CM) paradigm [3], proposed within the area of qualitative modeling [4]. CM is automated modeling paradigm used for automated building qualitative models from observations in presence of domain specific modeling knowledge. In the CM approaches, modeling knowledge is organized as a library of model fragments (equivalent to the models of individual processes in our formalism). Given a modeling task specification (or scenario, in the terminology of CM), CM composes a set of appropriate model fragments into a model that is suitable for modeling the observed system. The obtained model is evaluated by qualitative simulation. Although the concepts introduced within the CM area are relevant also for automated building of quantitative models of real world systems, this venue of work has not been explored much. A notable exception is the PRET reasoning system for automated modeling of dynamic systems [1]. PRET employs other kinds of knowledge in the process of automated modeling in form of "conservation rules". The example of such rule in electrical circuits domain is Kirchoff's law specifying that the sum of input and output currents at any observed point in the circuit is zero. Similarly, the force balance rule in mechanics domain specifies that the sum of forces at any observed coordinate of the mechanical system is zero. Thus, the knowledge used in PRET is much more general and cross-domain, as opposed to domain specific knowledge used in our approach.

The approach presented here follows our previous work on building process based models from measured data, presented in [2] and [5]. Note however, that the taxonomy of process classes introduced here allows for better organization of knowledge. Also, here we made the combining scheme explicit (i.e., declarative), as opposed to the implicit combining scheme used in [2] and assumption that models of individual processes are always additively combined in [5].

The usability of the approach should be further confirmed by applying it to the tasks of reconstructing dynamic system models from other domains. The ultimate goal of further and ongoing empirical evaluation is to apply the approach to different tasks of modeling real-world systems from measured data.

Although the scope of this paper is modeling dynamic change of the observed system through time, the approach can be extended to incorporate knowledge about spatial processes also. In the population dynamics domain, these processes would reflect the spatial diffusion of populations through the environment. The extended approach would allow modeling of spatio-temporal dynamic systems with partial differential equations (PDEs). The PADLES method [11] can be used for discovery of PDEs.

Acknowledgements

We acknowledge the support of the cInQ (Consortium on discovering knowledge with Inductive Queries) project, funded by the European Commission under contract IST-2000-26469.

References

1. E. Bradley, M. Easley, and R. Stolle. Reasoning about nonlinear system identification. *Artificial Intelligence*, 133:139–188, 2001.
2. S. Džeroski and L. Todorovski. Encoding and using domain knowledge on population dynamics for equation discovery. In L. Magnani and N. J. Nersessian, editors, *Logical and Computational Aspects of Model-Based Reasoning*, pages 227–248. Kluwer Academic Publishers, Boston, MA, 2002.
3. B. Falkenheiner and K. D. Forbus. Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, 51:95–143, 1991.
4. B. Kuipers. *Qualitative reasoning: modeling and simulation with incomplete knowledge.* MIT Press, Cambridge, MA, 1994.
5. P. Langley, J. Sanchez, L. Todorovski, and S. Džeroski. Inducing process models from continuous data. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 347–354, San Mateo, CA, 2002. Morgan Kaufmann.
6. P. Langley, H. A. Simon, G. L. Bradshaw, and J. M. Żythow. *Scientific Discovery.* MIT Press, Cambridge, MA, 1987.
7. L. Ljung. *System Identification - Theory For the User.* PTR Prentice Hall, Upper Saddle River, NJ, 2000. Second Edition.
8. J. D. Murray. *Mathematical Biology.* Springer, Berlin, 1993. Second, Corrected Edition.
9. L. Todorovski. *Using domain knowledge for automated modeling of dynamic systems with equation discovery.* PhD thesis, Faculty of computer and information science, University of Ljubljana, Ljubljana, Slovenia, 2003.
10. L. Todorovski and S. Džeroski. Declarative bias in equation discovery. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 376–384, San Mateo, CA, 1997. Morgan Kaufmann.
11. L. Todorovski, S. Džeroski, A. Srinivasan, J. Whiteley, and D. Gavaghan. Discovering the structure of partial differential equations from example behavior. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 991–998, San Mateo, CA, 2000. Morgan Kaufmann.
12. T. Washio and H. Motoda. Discovering admissible models of complex systems based on scale-types and identity constraints. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, volume 2, pages 810–817, San Mateo, CA, 1997. Morgan Kaufmann.