Discovering Dynamics: From Inductive Logic Programming To Machine Discovery *

Sašo Džeroski and Ljupčo Todorovski
Institut Jožef Stefan
Jamova 39, 61111 Ljubljana
Slovenia

Saso.Dzeroski@ijs.si, Ljupco.Todorovski@ijs.si

Abstract

Machine discovery systems help humans to find natural laws from collections of experimentally collected data. Most of the laws found by existing machine discovery systems describe static situations, where a physical system has reached equilibrium. In this paper, we consider the problem of discovering laws that govern the behavior of dynamic systems, i.e., systems that change their state over time. Based on ideas from inductive logic programming and machine discovery, we present two systems for discovery of qualitative and quantitative laws from quantitative (numerical) descriptions of dynamic system behavior.

1 Introduction

The task of modelling natural and artificial dynamic systems, i.e., systems that change their state over time, is omnipresent. To make the modelling task easier, qualitative modelling formalisms, such as QSIM, [Kuipers 1986] use qualitative relationships to describe dependencies among the system variables. In QSIM, these relationships are qualitative differential equations, in contrast to ordinary differential equations used to numerically model dynamic systems. States are also described qualitatively as pairs of qualitative values and qualitative directions of change.

Considerable effort has been devoted to the problem of automating the task of building qualitative models from example behaviors, i.e., the task of qualitative identification of dynamic systems [Coiera 1989], [Kraan et al. 1991]. Viewing qualitative models as logic programs and representing the QSIM theory in logic, [Bratko et al. 1992] and

*The work reported in this paper is supported by the Slovenian Ministry of Science and Technology and is part of the ESPRIT BRA Project 6020 Inductive Logic Programming.

[Džeroski and Bratko 1992] used systems for inductive synthesis of logic programs (inductive logic programming [Muggelton 1992]) to automatically synthetize qualitative models from example behaviors. MISQ [Kraan et al. 1991] can generate a qualitative model from a numerical trace by translating the numerical trace into a qualitative behavior from which a qualitative model is then generated.

The first part of the paper describes QMN (Qualitative Models from Numerical traces), a system that generates qualitative models (qualitative differential equations) from numerically described behaviors directly, without translating them to qualitative behaviors. To this end, QMN assumes the original numerical interpretation of the qualitative constraints used in QSIM, which include addition, multiplication and time derivation. A generate and test methodology is used in QMN, similar to the ones used in GENMODEL [Coiera 1989] and MISQ [Kraan et al. 1991].

It turns out that there is only a short step from the automatic generation of qualitative differential equations from numerical traces to the generation of ordinary differential equations from numerical traces. The latter problem is actually the problem of numerical discovery of the dynamics of a given system. Although there exists a variety of systems for numerical discovery, they have not addressed the problem of discovering dynamics from example behaviors. This problem is attacked by the discovery system LAGRANGE, described in the second part of the paper.

Initially, LAGRANGE was to use a very simple process: give the derivatives of the observed system variables and the system variables themselves, observed over a period of time, to an existing system for numerical discovery. The discovery system would then produce a set of laws (differential and algebraic equations) describing the behavior of the observed system. However, existing discovery systems were not suitable for use within LAGRANGE as they ask for additional data or consider dependencies between two variables only.

We thus had to develop a discovery mechanism to be used within LAGRANGE, which does not ask for additional data and considers dependencies among several variables at the same time. It is based on multidimensional linear regression, and introduces new terms by multiplication, following ideas from other discovery systems, such as BACON [Langley et al. 1987], ABACUS [Falkenheiner and Michalski 1990] and FAHRENHEIT [Żytkow and Zhu 1991], and inductive logic programming systems [Džeroski et al. 1992], [De Raedt and Bruynooghe 1993].

Section 2 first defines the task of numerical discovery of dynamics and then describes QMN, a system that generates qualitative models from numerically described behaviors. LAGRANGE, a machine discovery system that constructs quantitative models of dynamic systems, i.e., sets of algebraic and differential equations, is described in Section 3. Experimental evaluation of QMN and LAGRANGE is given in Section 4. Finally, Section 5 concludes with a discussion of related work and directions for further work.

2 The QMN algorithm

The task of identification of dynamic systems by numerical discovery is to find a set of laws that describe the dynamics of the system, given a sample behavior. More precisely, a set of real-valued system variables is measured at regular intervals over a period of time, as illustrated in Table 1. The laws to be discovered (also called a model of the dynamic system) typically take the form of a set of qualitative or ordinary differential equations.

Table 1: A behavior trace of a dynamic system.

time	X_1	X_2	...	X_n
t_0	x_{10}	x_{20}		x_{n0}
$t_0 + h$	x_{11}	x_{21}		x_{n1}
		...		
$t_0 + Nh$	x_{1N}	x_{2N}		x_{nN}

QMN (Qualitative Models from Numerical traces) generates a qualitative model from a numerically described dynamic system behavior. QSIM [Kuipers 1986] constraints on the system variables are used in the generated qualitative model. The constraints are taken from the repertoire given in Table 2.

Table 2: QSIM constraints tested in QMN.

QSIM Constraint	*Meaning*
$const(F)$	F *is constant over time*
$deriv(F1, F2)$	$F2$ *is time derivative of* $F1$
$minus(F1, F2)$	$F1 = -F2$
$M^+(F1, F2)$	$F1$ *is monotonically increasing with* $F2$
$M^-(F1, F2)$	$F1$ *is monotonically decreasing with* $F2$
$add(F1, F2, F3)$	$F1 + F2 = F3$
$mult(F1, F2, F3)$	$F1 * F2 = F3$

The input to QMN is a behavior trace of a dynamic system, such as the one given in Table 1. In addition, the values of three parameters have to be specified. These are: the order o of the dynamic system (the order of the highest derivative appearing in the dynamics equations), the maximum depth d of new variables introduced by combining old variables, and the error tolerance ϵ used when testing qualitative constraints.

Table 3 gives the QMN algorithm. Taking the set of system variables $S = \{X_1, \ldots, X_n\}$, QMN first introduces their time derivatives (up to order o). It then introduces new variables by repeatedly applying the basic arithmetic operations to the variables from S and their time derivatives. Finally, given the set of all (old and new) variables, it generates and tests all possible qualitative constraints.

The time derivatives of the system variables are introduced by numerical derivation in step 1. Step 2 combines all variables (the system variables and their time derivatives) pairwise in all possible ways, using the four basic arithmetic operations. If $v_i = |V_i|$, then we have $v_0 = (o+1)n$ and $v_i \leq 4v_0v_{i-1}$, which gives $v_{d-1} = \mathcal{O}((4n(o+1))^d)$ at the end of step 2. The values of the new variables in all time points are also calculated in this step.

Table 3: The QMN algorithm.

```
1. Introduce time derivatives (up to order $o$) of the system variables
   $D := \emptyset$
   for all variables $v$ in $S$ do
       $v_0 := v$
       for $i := 1$ to $o$ do
           $v_i := \frac{d}{dt} v_{i-1}$ (* $v_i = \dot{v}_{i-1}$ *)
           $D := D \cup \{v_i\}$
   $V := S \cup D$

2. Introduce new variables
   $V_0 := V$
   for $l := 1$ to $d - 1$ do
       $V_l := \emptyset$
       for all pairs of variables $(v, u) \in V_0 \times V_{l-1}$ do
           $a_{v,u} := v + u$
           $s_{v,u} := v - u$
           $m_{v,u} := v * u$
           $d_{v,u} := v/u$
           $V_l := V_l \cup \{a_{v,u}, s_{v,u}, m_{v,u}, d_{v,u}\}$
           (* duplicate terms removed in this step *)
       $V := V \cup V_l$

3. Generate and test qualitative constraints
   $M := \emptyset$
   for all $v \in V$ do
       test $const(v)$
       add to $M$ if holds
   for all $(v, u) \in V \times V$ do
       test $deriv/minus/M^+/M^-(v, u)$
       add to $M$ if holds
   for all $(v, u, w) \in V \times V \times V$ do
       test $add/mult(v, u, w)$
       add to $M$ if holds
```

The qualitative constraints are generated and tested in step 3. The total number of constraints tried is $\mathcal{O}({v_{d-1}}^3)$, that is $\mathcal{O}((4n(o+1))^{3d})$. This number is exponential in the parameter d.

The testing of generated constraints is done as follows:

- A system variable X is considered constant if its standard error is below the error tolerance ϵ, i.e., $const(X) \Leftrightarrow \sigma_X < \epsilon$.

- To test whether Y is a derivative of X, the numerical derivative of X, named $\dot{X}$, is first computed. A linear regression $Y = a\dot{X} + b$ is then performed. If the correlation coefficient R and a are close to one and b is close to zero, all within the error tolerance ϵ, then Y is considered a derivative of X.

- For the constraint $minus(X, Y)$ a linear regression $Y = aX + b$ is performed, and the constraint is considered to hold if R and a are close to minus one and b is close to zero, all within the error tolerance ϵ.

- For $add(X, Y, Z)$ and $mult(X, Y, Z)$, a linear regression $Z = aW + b$ is performed, where $W = X + Y$ and $W = XY$, respectively. The constraints are considered to hold if R and a are close to one and b is close to zero, all within the error tolerance ϵ.

- To test whether the constraint $M^+(X, Y)$ is consistent with the observed behavior, one has to check that for all time points t_1 and t_2 it holds that $(X_{t_1} < X_{t_2} \Rightarrow Y_{t_1} < Y_{t_2})$. Similarly, $M^-(X, Y)$ is consistent with the sample behavior if $\forall t_1, t_2 : (X_{t_1} < X_{t_2} \Rightarrow Y_{t_1} > Y_{t_2})$.

3 The LAGRANGE algorithm

While QMN generates a set of qualitative differential equations from a given behavior of a dynamic system, LAGRANGE is able to generate a set of ordinary (quantitative) differential equations. The input to LAGRANGE is a behavior trace of a dynamic system specified in Table 1. In addition, the values of four parameters have to be specified. These are: the order o of the dynamic system (the order of the highest derivative appearing in the dynamics equations), the maximum depth d of new terms introduced by combining old terms (variables), the maximum number r of independent regression variables used for generating equations, and the error tolerance ϵ used when testing equations.

Table 4 gives the LAGRANGE algorithm. Taking the set of system variables $S = \{X_1, \ldots, X_n\}$, LAGRANGE first introduces their time derivatives (up to order o). It then introduces new variables (terms) by repeatedly applying multiplication to variables from S and their time derivatives. Finally, given the set of all (old and new) variables, it generates and tests equations by using linear regression.

The time derivatives of the system variables are introduced by numerical derivation in step 1. Step 2 introduces as new variables all the terms of depth not greater than d consisting of system variables and their derivatives (variables from V_0). When introducing new variables, terms of depth l are gathered in V_{l-1}. A term $\prod_{i=1}^{n} X_i^{\alpha_i}$ is of depth l iff $l = \sum_{i=1}^{n} \alpha_i$. For example, X_1X_2 is of depth 2 and $X_1^3X_2^2$ is of depth 5. As V_0 contains $(o+1) \cdot n$ variables, V_l

Table 4: The LAGRANGE algorithm.

```
1. Introduce time derivatives (up to order $o$) of the system variables
   $D := \emptyset$
   for all variables $v$ in $S$ do
       $v_0 := v$
       for $i := 1$ to $o$ do
           $v_i := \frac{d}{dt} v_{i-1}$ (* $v_i = \dot{v}_{i-1}$ *)
           $D := D \cup \{v_i\}$
   $V := S \cup D$

2. Introduce new variables with multiplication
   $V_0 := V$
   for $l := 1$ to $d - 1$ do
       $V_l := \emptyset$
       for all pairs of variables $(v, u) \in V_0 \times V_{l-1}$ do
           $n_{v,u} := v * u$
           $V_l := V_l \cup \{n_{v,u}\}$
           (* duplicate terms removed in this step *)
       $V := V \cup V_l$

3. Do linear regression on subsets of all variables
   $M := \emptyset$
   for $i := 1$ to $r + 1$ do
       for all subsets $R \in \mathcal{P}(V)$, $|R| = i$ do
           choose dependent variable $y \in R$
           if Linear Regression $(y, R \setminus \{y\})$ is significant
           then $M := M \cup \{y = c_0 + \sum_{x \in R \setminus \{y\}} c_x x\}$
               (* add Linear Regression formula to $M$ *)
```

contains $\mathcal{O}(((o+1)n)^{l+1})$ variables. Consequently, at the end of step 2, $|V| = \mathcal{O}(((o+1)n)^d)$. The values of the new variables in all time points are also calculated in this step.

Equations are generated and tested in step 3. Roughly speaking, each subset of V sized at most $r+1$ is used to generate a linear equation, where one of the terms is expressed as a linear combination of the remaining ones. The constant coefficients in the linear equation are determined by applying linear regression [Volk 1958]. The multiple correlation coefficient R [Volk 1958] is used to judge the significance of the equation. If $R > 1 - \epsilon$, where ϵ is a prespecified threshold (a user definable parameter in LAGRANGE), the equation is considered significant and is retained in the model of the dynamic system.

The total number of regressions tried is $\mathcal{O}(|V|^{r+1})$, that is $\mathcal{O}(((o+1)n)^{d(r+1)})$. While this number is exponential in the parameters d and r, we should note that small values of these parameters were sufficient for all the experiments we performed ($d = 2$, $r = 3$).

4 Experimental evaluation

The experimental evaluation of QMN and LAGRANGE was done as follows: A set of differential equations, modelling a real-life dynamic system was first chosen, as well as appropriate values of the parameters involved. The initial state for the system variables, the integration step h for solving the differential equations and the number N of integration steps were then selected. The differential equations were then integrated using the fourth-order Runge-Kutta method [Press et al. 1986] (pp. 550–554). The obtained behavior was then given to QMN and/or LAGRANGE, which generated a set of laws describing the behavior.

Both QMN and LAGRANGE are implemented in the C programming language and were run on a Sun SPARC IPC workstation. All systems we considered were first-order dynamic systems. The parameters o and d were set to one in all experiments. The parameter r in LAGRANGE was set to three. The maximum running time for QMN and LAGRANGE for the experiments described was four seconds.

4.1 Experiments with QMN

QMN was applied to two dynamic systems, which have been used earlier as test examples for the automatic generation of qualitative models. These are a U-tube system and a cascaded tanks system [Kraan et al. 1991].

A large number of qualitative constraints were found to be true by QMN in both cases. Redundant and trivial constraints were removed manually. Future versions of QMN might benefit from a path-finding (graph connectedness) approach, similar to the one in MISQ [Kraan et al. 1991], which retains only those constraints necessary to connect all system variables.

U-tube

A U-tube system consists of two containers (A and B) connected by a thin pipe. Both containers have some water in them, one more than the other. The system is described by the following model

$$\begin{aligned}\dot{l_A} &= c(l_B - l_A)\\ \dot{l_B} &= -\dot{l_A}\end{aligned}$$

where l_A and l_B are the water levels in the tanks. The value $c = 2$ was chosen for the constant and the behavior of the system was simulated from the initial state $l_A = 100, l_B = 40$ for $N = 100$ steps of $h = 0.02$ time units.

The following qualitative constraints were found, among others, to be true for the U-tube system:

$$\begin{array}{ll} m\_plus(l_A, \dot{l_B}) & m\_plus(l_B, \dot{l_A}) \\ m\_minus(\dot{l_B}, l_B) & m\_minus(l_A, \dot{l_A}) \\ m\_minus(l_A, l_B) & \dot{l_A} = -\dot{l_B} \\ deriv(l_B, \dot{l_B}) & deriv(l_A, \dot{l_A}) \end{array}$$

Cascaded tanks

A cascaded tanks system has been used to illustrate the use of MISQ [Kraan et al. 1991] for learning qualitative models of dynamic systems. It consists of two tanks (A and B), where water flows from the first into the second. The first tank has a constant inflow. The whole system can be desoribed by the following model

$$\begin{array}{lllllll} i_A &=& c_1 & & & & \\ \dot{l_A} &=& i_A - o_A & \qquad & o_A &=& c_2\sqrt{l_A} \\ \dot{l_B} &=& o_A - o_B & \qquad & o_B &=& c_2\sqrt{l_B} \end{array}$$

where i_A is the inflow into tank A, o_A and o_B are the outflows from tanks A and B and l_A, l_B are the corresponding water levels. The values $c_1 = 200$ and $c_2 = 13$ were chosen for the constants and the behavior of the system was simulated from the initial state $l_{A0} = 10000, l_{B0} = 0$ for $N = 115$ steps of $h = 0.5$ time units.

Given the above behavior, the following qualitative constraints were found, among others, by QMN:

$$\begin{array}{ll} const(i_A) & \\ m\_minus(l_A, \dot{l_A}) & m\_minus(o_A, \dot{l_A}) \\ m\_plus(l_A, o_A) & m\_plus(l_B, o_B) \\ deriv(l_B, \dot{l_B}) & deriv(l_A, \dot{l_A}) \\ add(o_A, \dot{l_A}, i_A) & add(o_B, \dot{l_B}, o_A) \end{array}$$

4.2 Experiments with LAGRANGE

Models for several real-life dynamic systems were generated by LAGRANGE. These include a U-tube system, a cascaded tanks system [Kraan et al. 1991], a linear chemical reaction, and a predator-prey system [Babloyantz 1986].

A significant equation might be re-discovered several times in similar forms by LAGRANGE. For example, the equation $X + Y = Z$ could be re-discovered as $X^2 + XY = XZ$. Without loss of generality, we have removed this particular form of redundancies manually.

Future versions of LAGRANGE will take care of this problem automatically. More sophisticated redundancies are complicated to handle and have been left untouched, as in the chemical kinetics example.

U-tube

The following set of equations, equivalent to the original model, was generated by LAGRANGE from the U-tube system behavior.

$$\begin{aligned} l_B &= 140 - l_A \\ \dot{l}_A &= 280 - 4 \cdot l_A \\ \dot{l}_B &= -280 + 4 \cdot l_A \end{aligned}$$

This set of equations is redundant. Any single equation can be removed without loss of completeness.

Cascaded tanks

The following set of equations, equivalent to the original model, was generated by LAGRANGE for the cascaded tanks system described above.

$$\begin{aligned} i_A &= 200 & & \\ \dot{l}_A &= 200 - o_A & \qquad 169 \cdot l_A &= {o_A}^2 \\ \dot{l}_B &= o_A - o_B & \qquad 169 \cdot l_B &= {o_B}^2 \end{aligned}$$

Chemical kinetics

The following is a simple chemical kinetics model of a two-step chemical reaction, where a substance A is transformed into substance B, which is then transformed into substance C [Babloyantz 1986] (p. 37).

$$\begin{aligned} A &\xrightarrow{k_1} B \\ B &\xrightarrow{k_2} C \end{aligned}$$

The constants k_1 and k_2 are the corresponding reaction rates. The evolution of the concentrations of the substances involved in the reaction is described by the following equations:

$$\begin{aligned} \dot{A} &= -k_1 A \\ \dot{B} &= k_1 A - k_2 B \\ \dot{C} &= k_2 B \end{aligned}$$

Only substance A was initially present ($A_0 = 100$, $B_0 = C_0 = 0$). The above differential equations, with reactions rates $k_1 = 2$ and $k_2 = 3$, were integrated with a time step of 0.01 units for 500 steps. An equivalent model was generated by LAGRANGE:

$$
\begin{aligned}
\dot{A} &= -2 \cdot A \\
\dot{B} &= 2 \cdot A - 3 \cdot B \\
\dot{C} &= 3 \cdot B \\
(*)\ C &= 100 - A - B
\end{aligned}
$$

The equations for $\dot{A}, \dot{B}$ and $\dot{C}$ are obtained by filling in the values of the constants k_1 and k_2 in the original model. Equation $(*)$, on the other hand, is not found in the original model. It expresses the law of mass conservation for the particular case at hand. Any single of the four generated equations can be removed without losing equivalence to the original model.

Population dynamics

A Volterra-Lotka model of periodic behavior can be used to model the coexistence of prey and predator populations [Babloyantz 1986] (p. 145). For example, take the populations of lynxes and hares. The latter is a vegetarian, the former a carnivore that hunts hares. The lynx population must behave in such a manner that it must not eat all the hares, or its own species will disappear. The dynamics of this system is described by the following model

$$
\begin{aligned}
\dot{N}_1 &= k_1 N_1 - s N_1 N_2 \\
\dot{N}_2 &= s N_1 N_2 - k_2 N_2
\end{aligned}
$$

We chose the initial hare and lynx populations to be $N_{10} = 10$ and $N_{20} = 140$, and the parameters $s = 0.01$, $k_1 = 1.6$, $k_2 = 0.2$. We than integrated the above differential equations with $h = 0.05$ for $N = 240$ time steps. The following two equations were generated by LAGRANGE:

$$
\begin{aligned}
N_1 N_2 &= 160 \cdot N_1 - 100 \cdot \dot{N}_1 \\
N_1 N_2 &= 20 \cdot N_2 + 100 \cdot \dot{N}_2
\end{aligned}
$$

They can be obtained by substituting the values of the parameters in the original equations, then multiplying by $100 = 1/s$ and expressing $N_1 N_2$ from the resulting equations.

5 Discussion

QMN and LAGRANGE are inspired and benefit from ideas developed in the field of inductive logic programming. First of all, QMN addresses the problem of learning qualitative models that has also been tackled by the GOLEM [Bratko et al. 1992] and mFOIL [Džeroski and Bratko 1992] systems for inductive logic programming. Second, both QMN and LAGRANGE introduce new variables in the same way as new variables are introduced by determinate literals in LINUS [Džeroski et al. 1992]. Finally, they are also related to the work on clausal discovery [De Raedt and Bruynooghe 1993], where integrity constraints are systematically generated and tested with respect to a given database.

Besides QMN, there exist other systems (such as MISQ [Kraan et al. 1991]) that are able to generate qualitative models from numerical traces. Also, considerable effort has been spent on generating qualitative models from qualitative behaviors [Coiera 1989], [Bratko et al. 1992], [Džeroski and Bratko 1992]. While GENMODEL [Coiera 1989] and MISQ [Kraan et al. 1991] cannot introduce new variables in the model, QMN can. Inductive logic programming systems can also introduce new variables, but have problems because of the indeterminacy of the new variables. Namely, when adding two qualitative variables, the outcome is not uniquely determined. No such problems are present in QMN as the new variables introduced numerically have uniquely determined values. The introduction of new variables in QMN is based on the idea of introducing new determinate literals within the LINUS inductive logic programming system [Džeroski et al. 1992]. We consider QMN to be a stepping stone from the area of learning qualitative models, which includes inductive logic programming, to the world of machine discovery, where LAGRANGE is situated.

LAGRANGE is unique among machine discovery systems in its ability to discover laws that govern the behavior of dynamic systems. Although it might be regarded as a small extension of existing discovery algorithms, it can handle a whole new world of problems which cannot be handled by existing discovery systems. As the task of automated modelling of dynamic systems is omnipresent, LAGRANGE is potentially applicable to a wide variety of real-life problems.

However, several problems have to be addressed to facilitate practical applications of LAGRANGE. Most notably, measurement errors have to be taken into account during the process of searching for laws (differential and algebraic equations). Approaches used in existing discovery systems, such as FAHRENHEIT [Żytkow and Zhu 1991], might prove useful in this respect.

As mentioned in the introduction, our initial idea for LAGRANGE was to employ a transformation approach (such as used in LINUS [Lavrač et al. 1991], [Džeroski et al. 1992]) and transform the problem of discovering dynamic to an ordinary discovery problem as addressed by existing discovery systems. The transformation would add the corresponding time derivatives to the existing system variables. The two basic requirements for a machine discovery system to be used on the transformed discovery problem are:

- It must be able to find laws involving more than one variable from observational data only, i.e. without asking for additional experiments.

- It must be able to find a set of laws, rather than a single one, where all laws hold for the domain as a whole. It is not explicitly stated beforehand which system variables are dependent and which are independent.

Existing discovery systems, such as BACON [Langley et al. 1987], ABACUS [Falkenheiner and Michalski 1990] and FAHRENHEIT [Żytkow and Zhu 1991], may satisfy the first or second criterion, or even both criteria partially, but do not fully satisfy both criteria at the same time. New discovery mechanisms had to be developed for LAGRANGE, to meet the above criteria in full. LAGRANGE thus extends the current work on machine discovery in several directions.

The most important contribution of our work is the extension of the scope of machine discovery to dynamic systems. LAGRANGE is able to construct a set of differential and/or algebraic equations describing a given behavior of a dynamic system. In this way, it extends the scope of machine discovery systems from high-school physics ($F = ma$) to college physics ($F = m\ddot{x}$).

Another important extension is the ability to generate a set of laws (constraints) that hold in a domain as a whole from observational data only, i.e. without asking for additional experiments. This is in contrast with most machine discovery systems which keep some variables constant and ask the user (scientist) to vary the others. This may prove impossible in many cases, especially in the context of dynamic systems.

LAGRANGE was successfully used to generate models for several dynamic systems. In all cases, LAGRANGE was able to generate models equivalent to the original. Judging on the successful use of LAGRANGE for building models of dynamic systems and the possibilities for further improvements, we conclude that LAGRANGE is a promising step towards the application of machine discovery to complex dynamic systems.

References

[Babloyantz 1986]
Babloyantz, A. (1986). *Molecules, Dynamics, and Life.* John Wiley & Sons, Inc., New York.

[Bratko et al. 1992]
Bratko, I., Muggleton, S., and Varšek, A. (1992). Learning qualitative models of dynamic systems. In Muggleton, S., editor, *Inductive Logic Programming*, pages 437–452. Academic Press, London.

[Coiera 1989]
Coiera, E. (1989). Learning qualitative models from example behaviors. In *Proc. Third International Workshop on Qualitative Physics.* Stanford, California.

[De Raedt and Bruynooghe 1993]
De Raedt, L. and Bruynooghe, M. (1993). A theory of clausal discovery. In *Proc. Thirteenth International Joint Conference on Artificial Intelligence.* To appear.

[Džeroski and Bratko 1992]
Džeroski, S. and Bratko, I. (1992). Handling noise in inductive logic programming. In *Proc. Second International Workshop on Inductive Logic Programming.* Tokyo, Japan. ICOT TM-1182.

[Džeroski et al. 1992]
Džeroski, S., Muggleton, S., and Russell, S. (1992). PAC-learnability of determinate logic programs. In *Proc. Fifth ACM Workshop on Computational Learning Theory*, pages 128–135. ACM Press, New York, NY.

[Falkenheiner and Michalski 1990]
Falkenheiner, B. and Michalski, R. (1990). Integrating quantitative and qualitative discovery in the ABACUS system. In Kodratoff, Y. and Michalski, R., editors, *Machine Learning: An Artificial Intelligence Approach*, pages 153–190. Morgan Kaufmann, San Mateo, CA.

[Kraan et al. 1991]
Kraan, I., Richards, B., and Kuipers, B. (1991). Automatic abduction of qualitative models. In *Proc. Fifth International Workshop on Qualitative Physics.* Austin, Texas.

[Kuipers 1986]
Kuipers, B. (1986). Qualitative simulation. *Artificial Intelligence*, 29(3):289–338.

[Langley et al. 1987]
Langley, P., Simon, H., and Bradshaw, G. (1987). Heuristics for empirical discovery. In Bolc, L., editor, *Computational Models of Learning.* Springer, Berlin.

[Lavrač et al. 1991]
Lavrač, N., Džeroski, S., and Grobelnik, M. (1991). Learning nonrecursive definitions of relations with LINUS. In *Proc. Fifth European Working Session on Learning*, pages 265–281. Springer, Berlin.

[Muggelton 1992]
Muggleton, S., editor (1992). *Inductive Logic Programming.* Academic Press, London.

[Press et al. 1986]
Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1986). *Numerical Recipes.* Cambridge University Press, Cambridge, MA.

[Volk 1958]
Volk, W. (1958). *Applied Statistics for Engineers.* McGraw-Hill, New York, NY.

[Żytkow and Zhu 1991]
Żytkow, J. and Zhu, J. (1991). Application of empirical discovery in knowledge acquisition. In *Proc. Fifth European Working Session on Learning*, pages 101–117. Springer, Berlin.