

Learning Qualitative Models in the presence of Noise.

George M. Coghill
Department of Computing Science,
University of Aberdeen,
King's College, Aberdeen AB24 3UE.
gcoghill@csd.abdn.ac.uk

Simon M. Garrett and Ross D. King
Department of Computer Science,
University of Wales,
Aberystwyth SY23 3DB.
{smg, rdk}@aber.ac.uk.

August 29, 2003

Abstract

The ability to learn a model of a system from observations of the system and background knowledge is central to intelligence, and the automation of the process is a key research goal of Artificial Intelligence. We present a model-learning system, developed for application to scientific discovery problems, where the models are scientific hypotheses and the observations are experiments. The learning system, QOPH learns the *structural* relationships between the observed variables, known to be a hard problem. QOPH has been shown capable of learning models with hidden (unmeasured) variables, under different levels of noise, and from qualitative or quantitative input data.

1 Introduction

In the development of intelligent tools to aid in the process of Scientific Discovery, particularly in the construction of explanatory models, is an important goal of AI [10]; and qualitative modelling provides an ideal representation.

Bioinformatics is an ideal domain for applying this technology: the data are sparse (making it unsuitable for numerical techniques), they are noisy and they require the construction of models which will inevitably include unobserved variables. Work on constructing models of systems in molecular biology is in the early stages of development and so, given the above stated challenges any useful results emerging will be of tremendous practical value.

The ultimate goal in this scientific quest is the production of quantitative models; however, the discovery of suitable structural models (qualitative differential equations) can be the means of directing the scientist as to which experiments to carry out next in the [path] towards this goal.

In this paper we present QOPH a learning system which combines Inductive Logic Programming (ILP) with QSIM in order to construct qualitative models of physical and biological systems containing unmeasured variables.

The paper is organised as follows: In the next section we give an outline of ILP and QSIM as they are utilised in QOPH; and a review of related work. QOPH itself is presented in the following section, along with a description of the experiments carried out to test it; followed by a presentation of the results obtained. The biological application is then introduced; and in the final section the overall results are discussed.

2 Background

2.1 Qualitative Simulation

QSIM [24, 22, 23] is a constraint based qualitative simulation engine and utilises an equational representation which is an abstraction of *ordinary differential equations*. It is the most highly developed constraint based Qualitative Reasoning (QR) system available. In this section the main focus will be on the *pure* QSIM algorithm, i.e. the algorithm as originally described in [23].

In QSIM, each model consists of a set of variables linked together via a set of *constraints*, called a *qualitative differential equation* (QDE). Each variable consists of a $\langle qmag, qdir \rangle$ pair. Here, *qmag* is the qualitative magnitude of the variable. It has a quantity space of varying resolution consisting of alternating points (called landmark values) and intervals; typically the quantity space is divided into the regions $[-\infty \dots 0), [0], (0 \dots \infty]$, where infinity is treated as a value. A *qdir* is the qualitative rate of change of the variable, which has a fixed, three valued resolution (the three quantities being *inc*, for increasing; *dec*, for decreasing; and *std*, for steady). Each constraint has only one operation and is defined between two or three variables.

There are several kinds of constraint which can appear in a QSIM model. There are predicates, implemented as relations, representing the usual algebraic operations, of addition, multiplication, and sign inversion; plus a derivative predicate stating that one variable is the derivative of another.

One of the attractive features of QSIM is that it is designed to handle incompleteness in the knowledge of the model. The incompleteness here takes the form of a lack of knowledge concerning functional relations in the system. This situation is captured by the monotonic function constraints **M+** and **M-** between two variables, which declares that one variable monotonically increases (+) or decreases (-) with respect to another variable, covering families of relations.

The conjunction of qualitative relations models the relationships between a set of measured variables, plus a number of putative, unmeasured variables. There may be zero, one or more unmeasured variables, known as the model's *hidden variables*. Where there are sufficient hidden variables, the method described here can discover *hidden relations* that relate only hidden variables; this appears to be a novel feature of the learning system presented here.

2.2 Inductive Logic Programming (ILP)

The general model learning problem can be represented deductively as follows: if we term the observations (*evidence*) E , the background knowledge B , and the hypothesis to be learnt H , then given that:

$$B \not\models E \tag{1}$$

find a hypothesis H so that

$$B \wedge H \models E \tag{2}$$

Many possible solutions to this problem are possible, e.g. the trivial solutions of E , or $B \rightarrow E$. The problem is therefore how to restrict solutions to suitable ones. In abduction [18, 15] solutions are restricted to ground facts; in ILP more general solutions are allowed [29, 28], although there are still typically syntactic restrictions on what form solutions can take [36]. For most scientific discovery problems it is clear that ILP is advantageous, as we wish to learn general theories; and for similar reasons ILP is a sensible choice for learning QSIM models.

ILP is distinguished from other machine learning techniques in using first-order predicate logic (specifically logic programs) to represent background knowledge, observations, and hypotheses [27]. For problems such as learning QSIM models, ILP can, in some circumstances outperform all other forms of learning. We have previously applied machine learning and ILP to many scientific problems with success (e.g. [20, 12, 34]).

The learning of qualitative models from examples is a great challenge for current machine learning methods since the search space is very large. The problem is also interesting because the data are *positive only*, i.e. when identifying a system, nature only provides positive examples of states of the system, not examples the system can *not* be in. This hinders machine learning as there are no negative examples to restrict over-generalisation.

2.3 Related Work

Automated model construction is an important and growing area of research which has as a central aim the provision of appropriate models for scientific and industrial tasks. We restrict this review to methods directly related to learning QSIM models.

The earliest work on inductive learning of qualitative models was Coiera’s GENMODEL, the results of which first appeared in 1989 [8, 9]. GENMODEL has an effective strategy for using positive-only data (identified by Hau & Coiera [17] as form of relative least general generalisation (RLGG) [29]), though it can also make use of negative examples if they are available. However, Genmodel can not introduce or allow hidden variables, and the results produced are usually *overconstrained*; a feature of many systems for learning qualitative models. Hau updated GENMODEL [17] and showed that QSIM models are efficiently PAC learnable from the input data. The work was also notable for pointing out that dimensional analysis [2] can be considered as a form of directed negative example generation. Hau also demonstrated GENMODEL working on qualitative data generated from real-valued experimental data in an impressive manner, but this was limited by the need for all the variables to be known from the outset, which is unrealistic in many domains.

MISC [21], consisted of three stages: quantitative to qualitative data conversion; generating and testing all possible constraints (relations), and building models from the constraints. It made use of dimensional analysis to relieve the extent of the search on the variables [2], which meant that each variable needs to be associated to a particular ‘type’. Thus, certain relations such as a derivative relation between two variables both of the same type, can be *a priori* regarded as impossible.

The main limitations of MISC were that the models produced tended to be overconstrained, particularly when less than complete input information was given. Ramachandran’s follow up to MISC, MISC-RT [30], addressed the problem of model induction over multiple operating regions.

Bratko *et al.* [4] used Muggleton and Feng’s GOLEM (ILP) program [28] along with QSIM to produce a model of the U-tube system. This work was important since it was the first to show that hidden variables could be introduced into the final model. The method also has a number of awkward requirements that made its application a rather limited solution: it required the use of negative examples, the model found by their system was shown to be logically equivalent to the standard U-tube model, but it was not *physically* equivalent from a systems theory point of view, and was overconstrained.

Say and Kuru [31] reviewed the work in the field and presented an interesting new approach (QSI). This starts with correlations, and then iteratively introduces new variables, building a model and comparing the output of that model with the known states until a satisfactory model is found. QSI works for positive-only data and can introduce hidden variables.

QSI searches the induction lattice, or model space starting with no information about the types or dimensions of the variables, nor with any background information about the nature of those types. Say and Kuru used redundancy checks and hanging variable tests, however the final model presented in their paper was again overconstrained.

Two other lines of investigation are relevant here. Kay’s semi-quantitative SQUID program [19] uses quantitative data to form an envelope around the functional relations of a system — i.e. it focuses particularly on parameter estimation rather than model structure induction — and Todorovski *et al.* [35] have induced qualitative models directly from the real-valued data. Their approach models partial differential equations (PDEs) following up on earlier work by Dzeroski [14], called QMN which learns a QSIM model directly from *quantitative* data and builds on the LAGRANGE program [13]. Neither approach can discover hidden variables.

3 Learning with QOPH

This section summarises the method used in our learning experiments; the use of clean and noisy data for learning; the model learning methodology of the QOPH program; various learning constraints used to increase efficiency of operation and details of how experiments were run and the use of high-level components to enable scalable learning. Fig.1 the elements of QOPH.

Note that this section only provides an *overview* of the learning method; many of the details required to reproduce the experiments are omitted here, due to space constraints, but can be found in [16] along with details of, and results from, other benchmark systems. Here we seek merely to present the principles

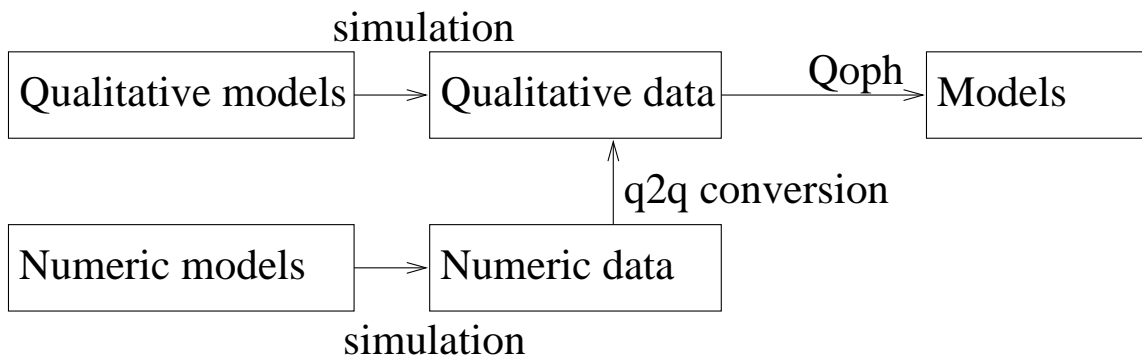


Figure 1: An outline of the inputs and outputs of the Qoph system

of the inductive method.

3.1 Model Learning Methodology - The QOPH Method

The ALEPH ILP system [33] was used as a wrapper for the QOPH implementation, which was written separately. As with [4], we used a subset of QSIM, implemented in Prolog, as background knowledge for ILP. The task of the model learning method is to induce a model given example values for a known set of qualitative variables (a set of qualitative states), and the model language of qualitative relations that can be applied to those variables.

ILP, like much of learning, can be considered to be a search through a space of possible solutions (e.g. [26], *p.23 ff.*). In the case of learning QSIM models, this space is the set of all possible QSIM models, partially ordered by generality. The relation-variable lattice is traversed by best-first search, and the search of this space can be constrained by the use of various heuristics. These heuristics can be generated from a number of sources: for example systems theory or the domain knowledge of the areas under investigation. In the former case the heuristics consist of general principles from systems theory such as: models must be parsimonious, operate under integral causality, and contain no algebraic loops (although these latter are preferences rather than absolute rules - since for some systems it is not always possible to achieve them). Also, for example if one is working in the biological area some of the domain knowledge may consist of a set of rules regarding legal chemical reactions that may take place.

The operation of the learning system, and how it utilises the heuristics, is set out in Fig.2. Clauses of relations are constructed until a clause can be shown to be a model of the qualitative states under the required conditions. A more detailed description of the components of the algorithm is contained in [16].

As Fig.2 indicates, models are built, via search, in an incremental manner, beginning with an empty model, and adding relations until a complete model is found. For each new relation added, a number of variables are selected from the list of previously seen variables. Initially, this list is the set of measured variables in the input data. A new, hidden variable may be introduced at this point. Once a hidden variable has been introduced in a relation it becomes available for use in other relations, and these other relations may in turn introduce other new hidden variables. In this way it is eventually possible, where required, to posit relations that relate only hidden variables. For reasons of parsimony however, relations are preferred that do not contain hidden variables.

3.2 Testing the QOPH method

The QOPH system was developed as a tool to aid in the construction of structural models of systems in molecular biology. This is a domain in which data are sparse and inherently noisy; therefore it was important that QOPH be thoroughly tested under these conditions in order to ascertain its potential as such a tool.

While it might be possible to generate a comprehensive set of noise conditions from numerical data it would be time consuming and tedious; whereas qualitative reasoning can provide a global picture of the ways in which a system may behave, contained within a finite set of qualitative states. Any qualitative state not contained within the environment for the system can be considered as a noise state; therefore

```

• qoph(Variables, RelPredicates, QualStates):
  - extendClause(Variables, RelPredicates,
    QualStates, {})

• extendClause(Variables, RelPredicates, QualStates,
  Clause):
  - Variables = Variables + HVar
  - LRList = generateLegalRels(RelPredicates,
    Variables)
  - SRList = sortRelsByCost(LRList)
  - foreach Rel in SRList
    * Clause = Clause + Rel
    * if ( full(Clause) and legal(Clause) and accurate(Clause, QualStates) ) then
      · print Clause
    * else
      · extendClause(Variables,
        RelPredicates, QualStates, Clause)
    * end if
    * remove Rel from Clause
  - end foreach

```

Figure 2: The QOPH algorithm

it is a straightforward task to test comprehensively the the ability of QOPH to learn in the presence of noise by either replacing systems states by noise states in the envisionment, or by adding noise states to the envisionment.

It is important to make clear that in the experiments to test the system, the term “noise” is used in different, though related, senses: noise inherent in the numerical data, noise introduced in the process of differentiation (required to obtain the *qdir* for the qualitative value), and ‘corruptions’ of the system states in the envisionment set. Each of these senses has relevance to the testing of the QOPH approach, therefore a complete set of experiments were developed to test all these aspects of the learning environment as follows:

1. Starting with a complete envisionment (containing N states) every combination of $N - K$ states from the envisionment (for $K = 0 \dots N$) was created (giving an experiment space of $2^N - 1$ experiments) and the ability of QOPH to learn the target model from each set of states was tested. This set of experiments measures the sensitivity of QOPH to sparcity of data alone.
2. For the complete envisionment of N states, experiments were run in which the total number of states used to by QOPH to learn from was kept constant (at N) with the number of real states being progressively replaced by a number of qualitative noisy states; from 0 (no noise) to N (only noise). A noisy qualitative state is defined as a state that is not part of the complete envisionment but is of the same form, containing the same number and type of variables. This tests the supposed effect of noise introduced in the quantitative to qualitative conversion process.
3. For a selection of the experiments used in (1) a random number of qualitative noisy states were added to the real ones and the effect on learning measured. This was done to simulate the effect of converting noisy signals.
4. Finally experiments were run in which the whole process (from data acquisition and interpretation to model construction) for both clean and noisy data were performed. The data conversion process for each case is described in detail in the following subsections.

We created both quantitative and qualitative versions of a set of *benchmark* systems. The qualitative form represents the system as a QDE, using elements of QSIM; the quantitative form of the same system is simply a parameterised numerical ODE version of the QDE model.

In order to illustrate the approach used and the results obtained we will utilise the well known coupled-tanks system. Details of the full set of tests and results for all benchmarks can be found in [16]. The input, $inflow_A$, pours into the top of tank A and the output, $outflow_B$, pours out of the base of tank B (see Fig. 3).

The model of this system is:

```

DERIV(levelA, netflowA),
DERIV(levelB, netflowB),
ADD(levelB, levelDiff, levelA),
M+(levelDiff, flowAB),
M+(levelB, outflowB),
ADD(netflowB, outflowB, flowAB),
ADD(flowAB, netflowA, inflowA).

```

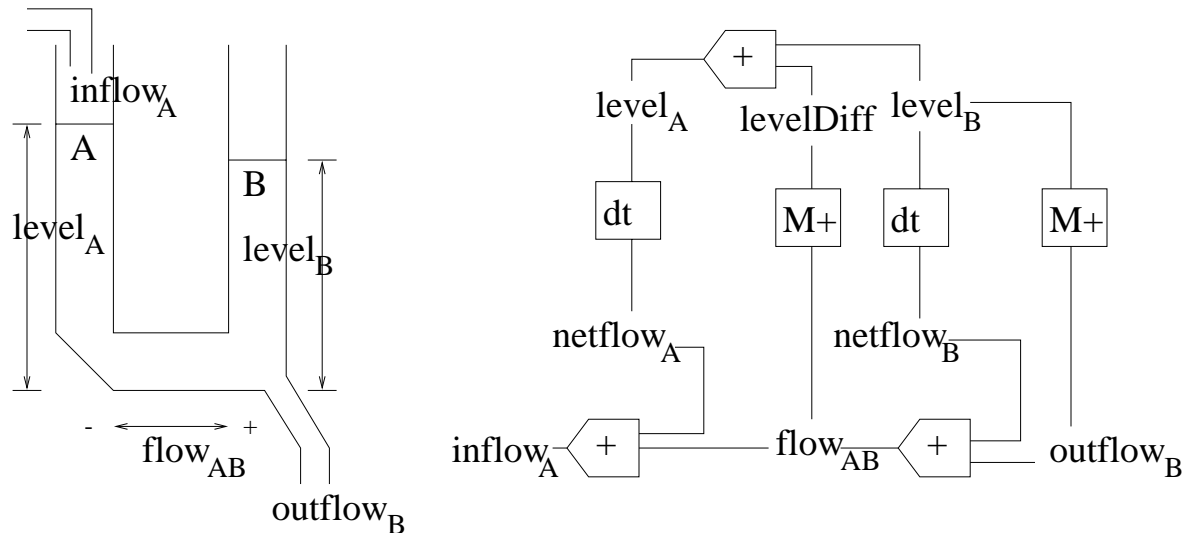


Figure 3: The coupled tanks (a) physical; (b) QSIM

Here there are three hidden variables, ‘ $netflow_A$ ’, ‘ $netflow_B$ ’, and ‘ $levelDiff$ ’. For the system to be correctly learned these variables will have to be induced. Variable ‘ $inflow_A$ ’ (the input) is exogenous to the model and so appears only once.

3.3 Creating the Clean Data

A complete envisionment was generated for a chosen input for each qualitative system. For *clean* data experiments, where the data contain no noise, either the entire complete envisionment or a subset of it formed the input data from which the model was to be learned.

One complication encountered was the very large number of experiments that needed to be performed to exhaustively explore the results of each subset. With systems such as the coupled tanks it is in general only possible to perform the experiments on a complete, physically meaningful *partition* of the state space, such as the envisionment for a single qualitative value for the exogenous variable(s). For this system it was possible to exhaustively examine one of the zero magnitude inputs, zero and steady $\langle 0, std \rangle$, which resulted in 10 unique states and therefore 2^{10} experiments.

For each experiment, the approach was to give QOPH the chosen set of states as its positive examples and a logic program that implements the QSIM relations described above as its background information.

To use our qualitative learning approach with numerical data required conversion from real-valued data to qualitative data. This is an important area of research in itself (e.g. [11, 5, 6, 1]). We employed a relatively simple but robust method. The N real-valued time series steps for a variable x were numerically

differentiated by means of a central difference approach [32] such that,

$$\left. \begin{aligned} \frac{dx_i}{dt} &= \frac{(x_i - x_{i-1}) + (x_{i+1} - x_i)}{2} \\ \frac{d^2x_i}{dt^2} &= (x_i - x_{i-1}) - (x_{i+1} - x_i) \end{aligned} \right\}_{i=2}^{N-1},$$

then the first and second derivatives were smoothed by applying a Blackman filter [3] to their Fast Fourier Transforms (FFT) and taking the real part of the inverse FFT. Smoothing is required because noise is introduced by the process of quasi-differentiation and cannot be avoided. Furthermore, temporal misalignments between two variables can occur during this process, introducing further errors.

In principle, having obtained values for x , $\frac{dx}{dt}$ and $\frac{d^2x}{dt^2}$ the values would just be converted to a qualitative quantity space such that:

$$\begin{aligned} x_i < 0 &\Rightarrow [x_i] = - \\ x_i = 0 &\Rightarrow [x_i] = 0 \\ x_i > 0 &\Rightarrow [x_i] = + \end{aligned} \tag{3}$$

in practice, however, a small margin of signal error,

$$\begin{aligned} (x_t < 0) \wedge (3\%.min(\mathbf{x}) \leq x_t \leq 0), &\Rightarrow x_t \mapsto 0 \\ (x_t > 0) \wedge (0 \leq x_t \leq 3\%.max(\mathbf{x})), &\Rightarrow x_t \mapsto 0 \end{aligned} \tag{4}$$

was allowed around the zero value since real values are highly unlikely to be exactly zero; this *zero envelope*, Z , allows us to assign near-zero real values to the qualitative zero quantity. Again there are errors introduced by this procedure since the signal may enter and leave the zero envelope any number of times, apparently jumping to and from zero. This may or may not be the desired behaviour: if the signal remains close to the edge of the envelope, small perturbations in the signal can cause errors in assigning the qualitative value.

For the numerical simulation, a mixture of zero and positive initial condition magnitudes were chosen for the two state variables; in the case of the coupled tanks with zero input, these were: (2,0), (0,3) and (2,3)¹. The numerical models produced data from these initial conditions until a steady state was approached. The data produced were then converted to qualitative states in the manner just described, and attempts were made to learn a model of the data. Experiments were also performed to learn models from the union of the qualitative states produced from the various different initial conditions.

3.4 Creating the Noisy Data

In order to be useful, the inductive system described here should be able to learn from noisy data. Noise was added to qualitative data by taking the complete envisionment, or a subset of it (as above) and a number of noisy qualitative states.

Noise was added to the numerical data as follows. The same sets of initial conditions were used and the resulting signals were modulated with varying amounts of Gaussian noise. The degree of noise was described as a fraction of a noise vector: thus we applied 1/1000, 10/1000, 100/1000, and 1000/1000 of the noise to the signal. Fig. 4 shows examples of the degrees of noise added to the coupled tanks signal given initial conditions of $level_A = 2$ and $level_B = 3$.

Experiments were performed for low magnitude noise first, for each of the systems, under all four inputs conditions. The noise was increased until learning was no longer achievable. The combined signal and noise were denoised by again applying a Blackman filter [3] to remove the high frequencies from the FFT domain, performing the inverse FFT and keeping the real part. This remaining signal was then converted to qualitative data (as above), giving an errant subset of the complete envisionment (ideally one full behaviour) for each numerical simulation, and experiments were run to learn the model from these data.

Interestingly, in almost all cases some states were missing from the full behaviour and errant states were added due to the unavoidably suboptimal conversion process and/or added noise; *clearly learning*

¹These values are given with respect to our numerical model, the parameterisation of which was arbitrary

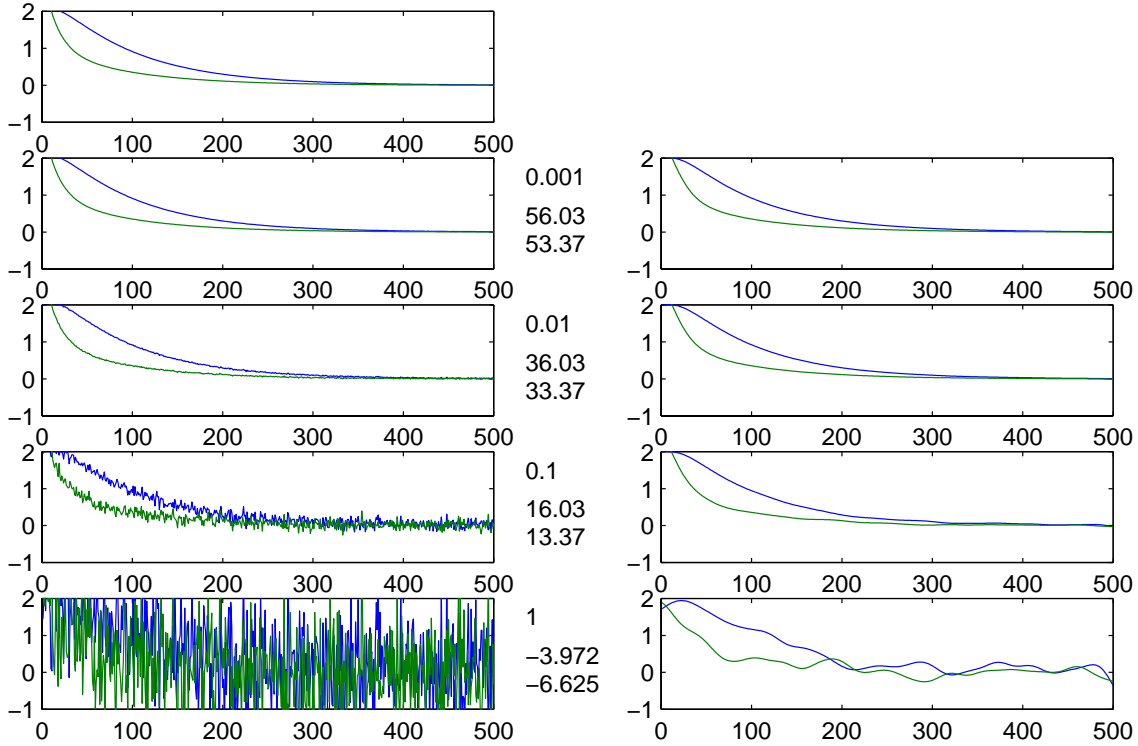


Figure 4: Graphs of levels of noisy (lhs) and smoothed (rhs) numerical data for the coupled tanks numerical model. Initial values: levelA=2 and levelB=3. The sets of three values in the centre refer to each row; the top figure is the noise scalar (0.001, 0.01, 0.1 and 1 of the raw noise), the other two values are the signal to noise ratio for the two levels

under these circumstances is very demanding. It is important to clarify that we set out to demonstrate the possibility of learning under these conditions using QOPH; in the future it will be possible to improve on our basic qualitative to quantitative approach, which in any case is not the focus of this work.

Finally, whilst it is possible to smooth data by applying a suitable Blackman filter (or something similar) — both for cleaning up the derivatives and removing noise — it should be pointed out that this is only meaningful where a large number of time steps are present, and in many applications this is not possible. Here, however, it suffices to demonstrate the principles being set out.

3.5 Experiments

The setup described above leads to the following sets of experiments. A given experiment induces zero, one or more models. A model was regarded as correct if it can be shown to be logically equivalent to a known correct *gold standard* clause. To simplify analysis, we restricted the number of model clauses produced after the induction routine to the first ten produced. If an isomer² of the appropriate gold standard were *not* among the induced clauses for a particular system's experiments, the test was regarded as a failure. In many cases the correct model would have been found if the search had been allowed to proceed beyond ten clausal models, but we wished to build a system that would generally find only the correct model, and find it quickly.

If some of the ≤ 10 clauses were found to be correct models this was regarded as a partial success. The ratio of the number of *correct* models to the total number of models induced is the *reliability* of the experiment. This measure of reliability were used in the results of both the clean and noisy data experiments. If an experiment produced only isomers of the gold standard then this is said to *reliably*

²In this context an isomer is a model that does not contain exactly the same set of constraints as the gold standard, but is logically equivalent. For example, two models which were identical except for the fact that one contained the constraint ADD(A, B, C) and the other the constraint SUB(C, B, A), would be isomers.

produce the correct model; another less reliable experiment may successfully induce the model as well as a number of non-equivalent models, or not find the correct model at all.

4 Results

The full results are too lengthy to be included here but can be obtained on request from the authors or from: <http://users.aber.ac.uk/smg/BISI/IE4/results.htm>.

Since any given experiment will induce its models from a finite number of states, it is possible to plot the *average* reliability for all the experiments for a particular number of states, from one state up to the number of states in the complete environment. This ‘Average reliability’ is given in the range [0 1]. For the noise experiments, the noise dimension is projected on the comparative 2-D plot (this assumes an average noise for each point on the state dimension) to allow comparison with clean data experiments, but a 3-D plot is also presented for the noise experiments that includes the noise dimension.

4.1 Models Learnt from Qualitative Data

The plots of the number of states against average reliability for the coupled tanks are shown in Fig.5.

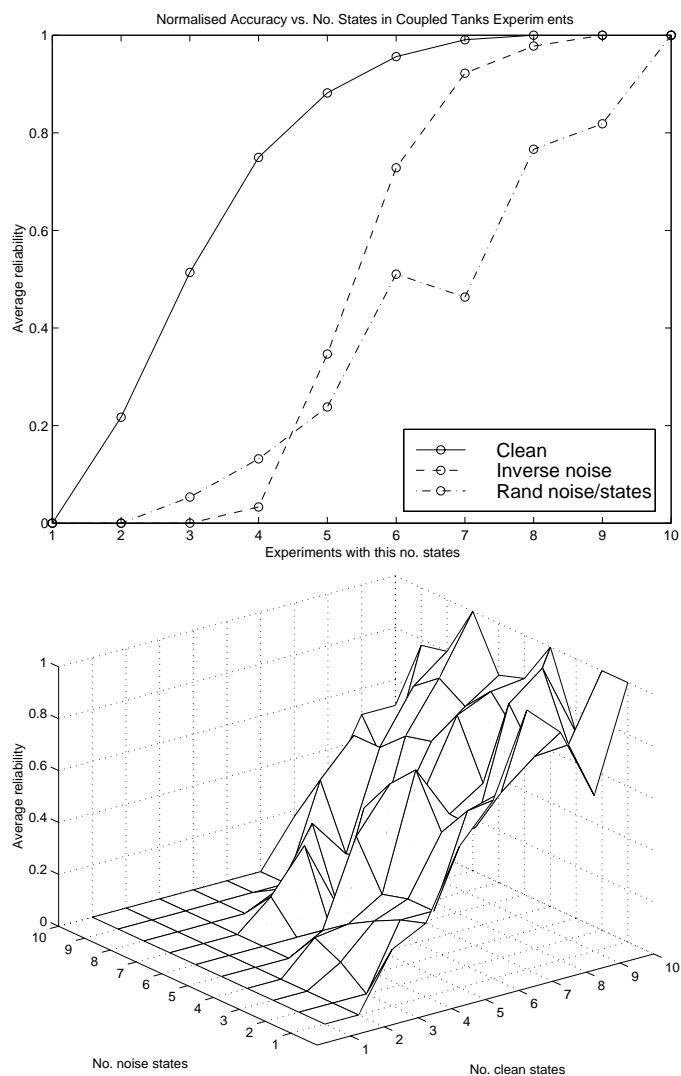


Figure 5: Coupled tanks reliability graphs: comparative 2D plots (left); state vs. noise vs. reliability (right)

State	$level_A$	$level_B$	$crossflow_{AB}$	$outflow_B$
0	$\langle 0, std \rangle$	$\langle 0, std \rangle$	$\langle 0, std \rangle$	$\langle 0, std \rangle$
1	$\langle 0, inc \rangle$	$\langle (0, \infty), dec \rangle$	$\langle (-\infty, 0), inc \rangle$	$\langle (0, \infty), dec \rangle$
2	$\langle (0, \infty), dec \rangle$	$\langle 0, inc \rangle$	$\langle (0, \infty), dec \rangle$	$\langle 0, inc \rangle$
3	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), dec \rangle$
4	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), std \rangle$	$\langle (0, \infty), dec \rangle$
5	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), inc \rangle$	$\langle (0, \infty), dec \rangle$
6	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), std \rangle$	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), std \rangle$
7	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), inc \rangle$	$\langle (0, \infty), dec \rangle$	$\langle (0, \infty), inc \rangle$
8	$\langle (0, \infty), std \rangle$	$\langle (0, \infty), dec \rangle$	$\langle 0, inc \rangle$	$\langle (0, \infty), dec \rangle$
9	$\langle (0, \infty), inc \rangle$	$\langle (0, \infty), dec \rangle$	$\langle (-\infty, 0), inc \rangle$	$\langle (0, \infty), dec \rangle$

Table 1: The environment states for the coupled tanks.

We analysed the performance of subsets of the complete environment to test whether certain subsets helped QOPH to learn the correct model more reliably than others. If this were the case then there would be a number of *minimal subsets* that contained the lowest number of states that reliably lead to the correct model being found; we label this set of minimal subsets S^+ . Subset analysis of the clean data experiments for the coupled tanks give the following states as the minimal subsets. Note that there are five elements in S^+ containing 2 states and 12 S^+ elements containing 3 states.

- [1,6], [6,8], [6,9] (state 6 with states 1, 8 and 9)
- [2,8], ([6,8]), [7,8] (state 8 with states 2, 6 and 7)
- [1,2,3], [1,2,4], [1,2,5] (states 1 and 2 with states 3, 4 and 5)
- [1,3,7], [1,4,7], [1,5,7] (states 1 and 7 with states 3, 4 and 5)
- [3,7,9], [4,7,9], [5,7,9] (states 2 and 9 with states 3, 4 and 5)
- [2,3,9], [2,4,9], [2,5,9] (states 7 and 9 with states 3, 4 and 5)

Fig. 6 shows the relationship of these states in the environment graph. A comparison with Table 1 reveals two key features: a selection of states from different behaviours and the use of the critical points of the system are the key to inducing the correct model reliably (see Discussion section below).

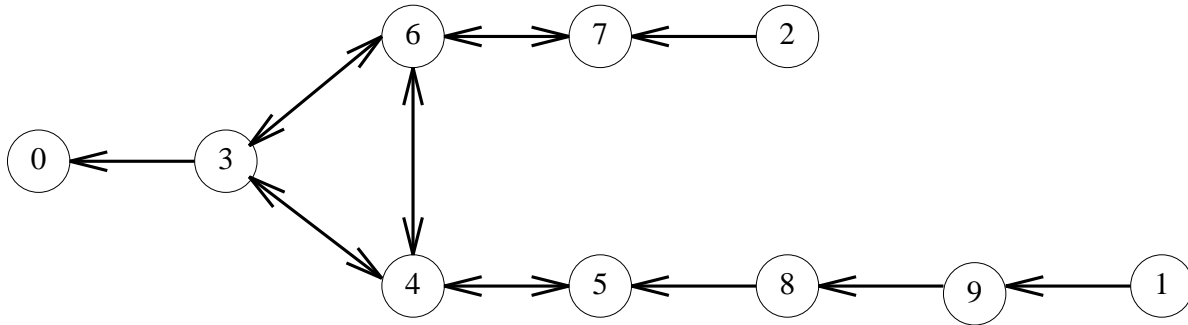


Figure 6: The environment graph for the coupled tanks

4.2 Benchmark Models Learnt from Numerical Data

The results from the numerical data experiments are presented in Fig. 7. The legend in the top right corner associates initial values of the state variables (given as two concatenated digits) to a plot; ‘all’ is the case where the union of states from all initial conditions were used in learning. These results show that it is possible to learn models from clean and noisy numerical data. As discussed above, the qualitative states generated from the clean numerical data contain a number of unavoidable data transformation errors, and the resulting qualitative states form at most a single behaviour of the system under investigation. The set of states gleaned from quantitative to qualitative conversion did not form a full behaviour for

either the coupled tanks or the spring, which makes the ability to learn a model from them even more impressive.

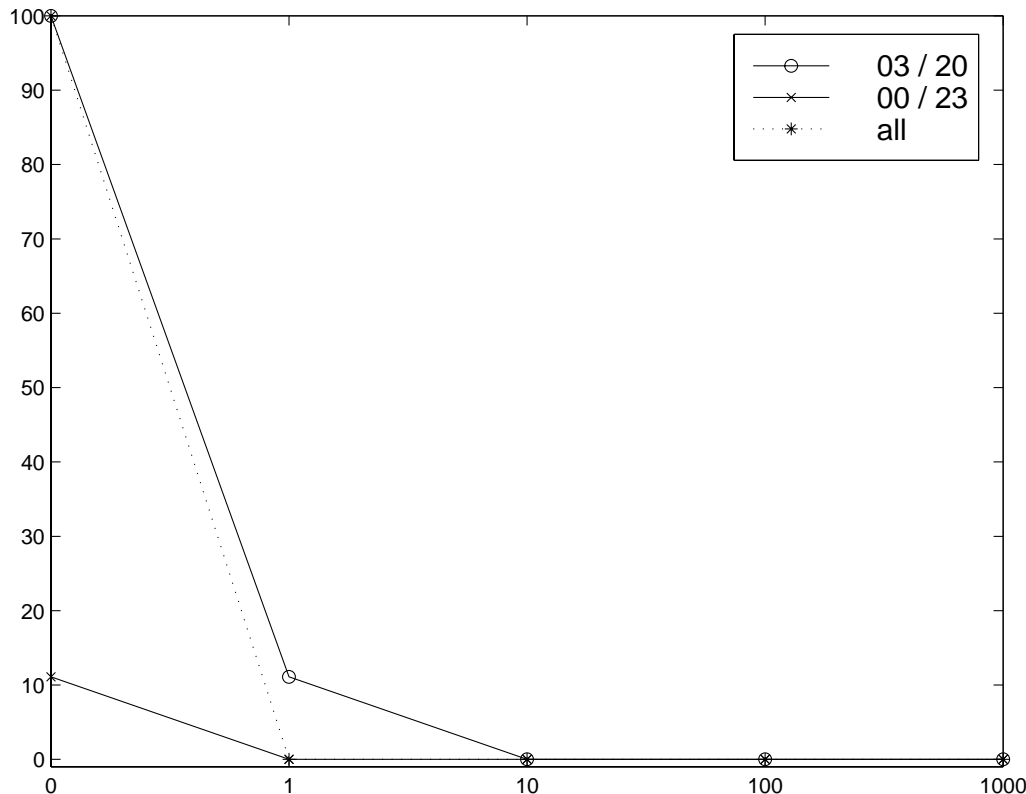


Figure 7: Reliability of learning the correct model from numerical data vs. 1000ths of full Gaussian noise for coupled tanks

5 Application to Biological Systems

5.1 Metabolic Components

As well as exploring the effects of sparsity of data and adding noise it was important to test the *scalability* of the QOPH learning method. So far we have only described models constructed from the basic QSIM primitives; to improve scalability it was useful to be able to use the well-established AI principle of *chunking* [25].

Metabolic pathways essentially contain only two types of molecule: metabolites and enzymes, we therefore designed two *Metabolic Components*, built from standard QSIM relations, to model *metabolites* and *enzymes*. Concentrations of metabolites vary over time as they are synthesised or utilised by enzymatically catalysed reactions. This means that their concentration at time t is a function of their concentration at time $t - 1$, and the amount that they are used or created by various enzyme reactions. This can be expressed as a simple summation in QSIM. The qualitative equation for the metabolite components is therefore:

$$\frac{dM}{dt} = M(t) + \sum_{i=0}^n (enzm_flow_i). \quad (5)$$

The other form of high-level metabolic component in a metabolic pathway are enzymes. Each enzyme is assumed to have one or two inputs and one or two outputs. If there are two inputs or outputs these are considered to form an input or output complex, such that the amount of the complex is proportional to the amount of the inputs or outputs multiplied together. This qualitatively models the probability that

both the inputs (or outputs) will collide with the enzyme with sufficient timeliness to be catalysed into the output complex (or input complex). The input complex is converted into the output complex which then disassociates into the output metabolites, and vice versa. The overall flow through the enzyme is the amount of input complex formed minus the amount of output complex formed. The qualitative equation for the enzyme components is therefore³:

$$flow = \underbrace{\mathbf{M}^+ \left(\prod_{i=1}^n M_i \right)}_{input\ complex} - \underbrace{\mathbf{M}^+ \left(\prod_{j=1}^m M_j \right)}_{output\ complex}. \quad (6)$$

This is an abstraction of standard kinetic equations [7] and is an expression of the collision probabilities of the metabolites and enzyme. We assume for simplicity that enzymes are taken to exist in constant amounts; although this is clearly a simplification this assumption is also used in ODE modelling. These metabolic components are shown in Fig. 8.

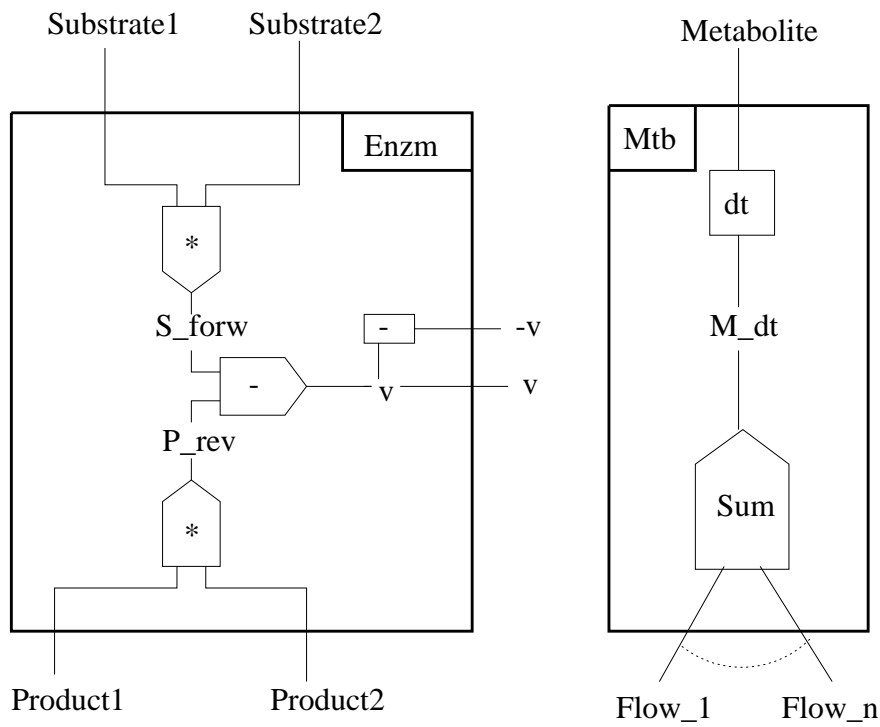


Figure 8: Metabolic components for metabolic system modelling as described in the text: ‘Enzyme’ (left) and ‘Metab’ (right)

Having established and justified the nature of the metabolic components, they can now be treated as standard modelling elements (just as much as standard QSIM constraints such as **ADD** or **M+**), permitting models to be built from them. We chose to model the glycolysis metabolic pathway using only the **METAB** (metabolite) and **ENZYME** (enzyme) relations between the variables, to generate its complete envisionment and to learn the model from those states.

5.2 Glycolysis Model Learnt from Qualitative Data

A model of glycolysis in *Trypanosoma brucei* constructed from Metabolic Components is shown in Fig. 9. The qualitative model is easier to understand than an ODE since it extracts out detail and allows a complete envisionment of the states.

³ Note the distinction between M and \mathbf{M}^+ , the amount of a metabolite and the monotonically increasing relation respectively.

We gave QOPH the states for the top part of this system, as shown below. There are 77 distinct legal qualitative states for this part of system.

```
glyc_top:-
  Glc = 1:0...inf/std,

  % Hexokinase
  enzyme([Glc,ATP], [F16BP,ADP],HKInvFlow,HKFlow),
  metab(1:F16BP, [f:HKFlow, f:AldInvFlow]),

  % Glycosal myokinase
  enzyme([ADP,ADP], [ATP,AMP],PFlow, PInvFlow),
  metab(1:ATP, [f:HKInvFlow, f:PFlow]), % ATP DEFINED
  metab(1:ADP, [f:HKFlow, f:PInvFlow]), % ADP DEFINED
  metab(1:AMP, [f:HKFlow]), % AMP DEFINED

  % Aldolase
  enzyme([1:F16BP], [1:DHAP,1:G3P], f:AldInvFlow,
    f:AldFlow),
  enzyme(1:DHAP, 1:G3P, f:TrioseInvFlow, f:TrioseFlow),
  metab(1:DHAP, f:DHdt, [f:AldFlow, f:TrioseInvFlow]),
  metab(1:G3P, f:G3dt, [f:AldFlow, f:TrioseFlow]).
```

The simplification of the first three steps in standard glycolysis proceeds on the basis that in a qualitative setting the conversion from glucose to glucose-6-phosphate, and from fructose-6-phosphate to fructose-1,6-bisphosphate both use ATP and produce ADP, and that the glucose-6-phosphate to fructose-6-phosphate is an isomorphic reaction; thus these three steps can be compressed into the single “hexokinase” step, as shown above, without compromising accuracy.

Having obtained this model it can be run to produce its complete envisionment - the full set of legal states that the model can be in, for all possible inputs. This complete envisionment can then be used as the data for learning the model using the method described above and in [16].

Using just the states from the model and the metabolic components, described above, and assuming that no knowledge about chemical interactions between the molecules in glycolysis, it was possible to learn the top part of the model in only a few hours.

Learning a model of this sort represents a major step forward because the learned system is equivalent to a QSIM model consisting of 36 relations, It was calculated that introducing high level components has more than doubled the complexity of the models that can be learned, as well as making the resulting models easier to read.

6 Discussion and Conclusion

Research in the area of learning models from data has established that models of different types of dynamic system can be learned. We have sought to build on the work reported by looking more closely at the classes of model that can be learned, and under what circumstances. Specifically, we have explored the robustness of the learning to two variations in the learning environment: reduction of the number of states (from the complete envisionment) used to learn the target model; the the introduction of “noisy” states into the data from which the models were to be learned. The results of performing a comprehensive set of experiments on benchmark models were presented.

The first general point is that for all the experiments the number of measured variables from which learning took place remained constant and was less than the total number of variables in the target model. Thus in all circumstances the learning system had to find the hidden variables and their relationships to the other variables of the model.

Analysis of the clean data experiments showed that given the complete envisionment of a system the correct model was always reliably found. As one would expect there was a gradual deterioration in the reliability as the number of states presented as data was reduced. However, a closer analysis of the results in conjunction with the envisionment graphs for the target models reveals that there is a strong relationship between the reliability of the learning process and the number, and type, of states used in an experiment.

The envisionment graph of the coupled tanks system has two main branches, the extremæ of which represent the case where one tank is full and the other empty (and vice versa). Looking now at the two element members of \mathcal{S}^+ we see that in each member the states making up the pair come from different branches of the envisionment graph; so we can hypothesise first of all that in order to reliably learn a

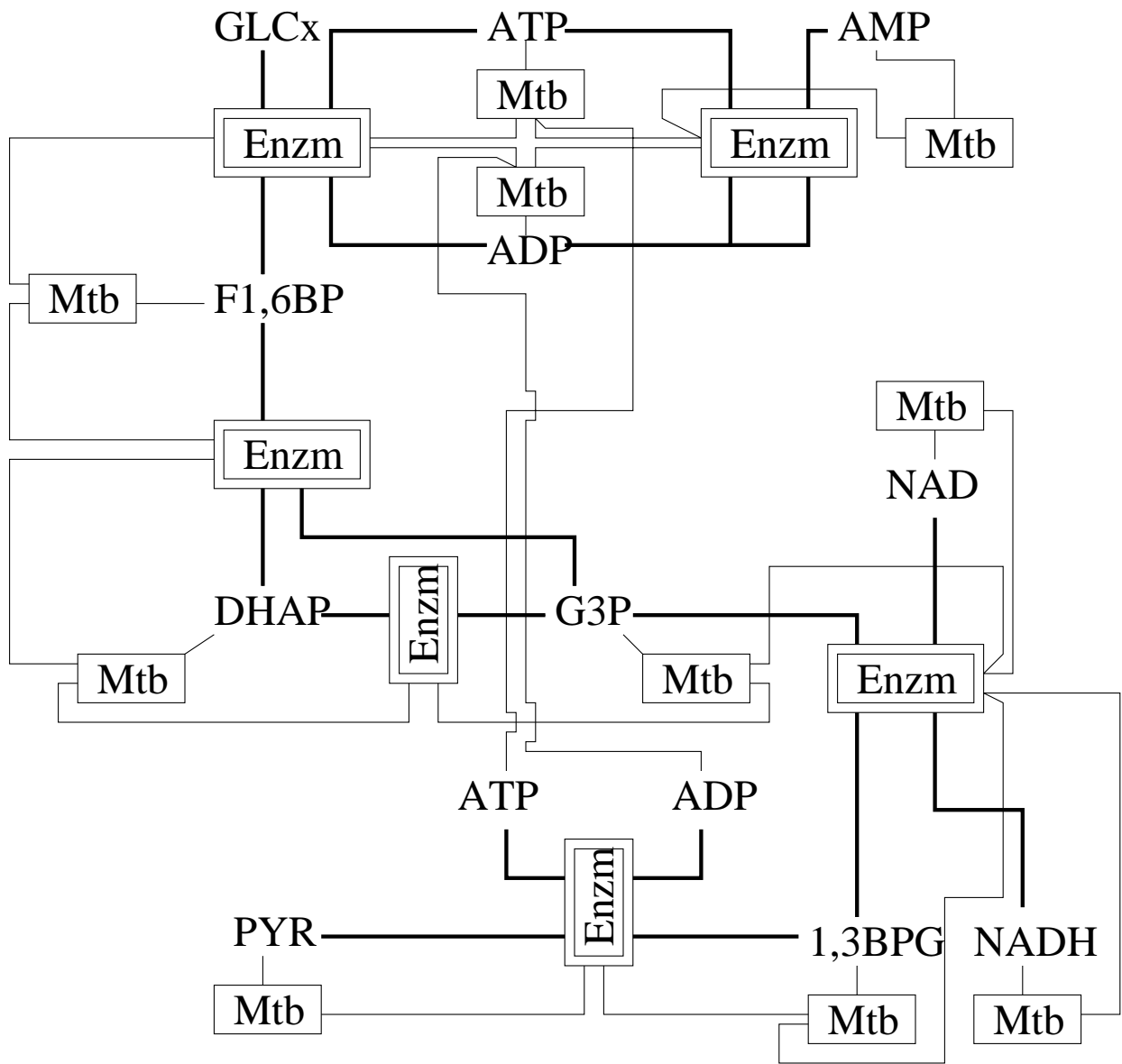


Figure 9: The glycolysis metabolic pathway, built from metabolic components

system the data used should come from experiments yielding qualitatively different behaviours (that is behaviours which would appear as distinct branches in an envisionment graph).

However, this hypothesis only provides a necessary, but not a sufficient condition for learning. It was noted in presenting the results that the key states in this result were states ‘6’ and ‘8’. These states represent critical points of the first derivative of the state variables which indicates the importance of these critical point states to the definition of a system. What this means is that if an experiment were set up in which all the state variables were exactly at their critical points then the experiment could be run for a very short time and the correct model structure identified. Of course, it is impossible to set up such an experiment, especially in the situation where the structure of the system is completely unknown. Another alternative is to set up multiple experiments with the state variables set to their extremæ; from which initial conditions all the states of the envisionment will eventually be passed through. The downside of this is that the experiments may be difficult to set up and could take an very long time to complete. These two scenarios form the ends of a spectrum within which the optimal experimental setting will lie; the identification of the the best strategies is an important area of research arising from the results of the present work, but it is beyond the scope of this paper.

Being able to model metabolic systems, using high level metabolic components, is in itself a useful tool since one can quickly simulate various scenarios without needing to parameterise the model; however, of more interest perhaps are the results of the glycolysis *learning* experiments. These show that the use of higher level components makes learning complex systems a practical possibility; something that until recently was not possible.

A summary of the results are:

- The benchmark models could be induced from their complete envisionments.
- As the number of states chosen from the complete envisionment increases so does the frequency and reliability of finding the correct model.
- The correct model can always be reliably found given a relatively small subset of the total envisionment. There is a set of these subsets such that other state subsets are either supersets of one member of this set, or do not reliably give rise to the correct model.
- Even though subsets containing *very few* states can reliably give rise to the correct model, it is possible to select subsets containing *almost all* the states that do *not* reliably lead to the correct model.
- Models can be learned from noisy simulated real data for the benchmark systems.
- Complex metabolic systems can be learned from qualitative data.

6.1 Future Work

Different search algorithms and heuristics could be explored for further speed benefits. For example, the use of a GP to search the model space may be faster, although there are other implementational difficulties with this approach. If causal ordering is always to be used, i.e. variables are only introduced where they can be fully constrained, then it would be more efficient to make it part of the model construction process; since causal ordering is not always possible our current approach is more flexible.

Having validated the model induction method on noisy qualitative and numerical data and demonstrated its ability to learn complex systems, the next step is to explore how successful it can be at modelling real data. To this end we are developing a method to learn models from metabolic and then transcriptomic microarray data.

Acknowledgement

This work is supported by BBSRC/EPSRC grant BIO10479. The authors would like to thank Stephen Oliver and Douglas Kell for some useful discussions on the biological aspects of this chapter. Early investigative work by Ashwin Srinivasan, with the authors, has also been most useful.

References

- [1] B. R. Bakshi and G. Stephanopoulos. Representation of process trends – part 3: multiscale extraction of trends from process data. *Computers Chem. Engng.*, 18:267–302, 1994.
- [2] R. Bhaskhar and A. Nigam. Qualitative physics using dimensional analysis. *Artificial Intelligence*, 45:73–111, 1990.
- [3] R. B. Blackman and J. W. Tukey. *The Measurement of Power Spectra*. John Wiley and Sons, New York, 1958.
- [4] I. Bratko and S. Muggleton. Learning qualitative models of dynamic systems. In S. Muggleton, editor, *Inductive Logic Programming*, pages 437–452. Academic Press, 1992.
- [5] J. T.-Y. Cheung and G. Stephanopoulos. Representation of process trends – part 1: a formal representation framework. *Computers Chem. Engng.*, 14:495–510, 1990.
- [6] J. T.-Y. Cheung and G. Stephanopoulos. Representation of process trends – part 2: the problem of scale and qualitative scaling. *Computers Chem. Engng.*, 14:511–539, 1990.
- [7] W. W. Cleland. The kinetics of enzyme-catalysed reactions with two or more substrates and products: 1. nomenclature and rate equations. *Biochim. Biophys. Acta*, 67:104–137, 1963.
- [8] E. W. Coiera. Generating qualitative models from example behaviours. Technical Report 8901, University of New South Wales, Department of Computer Science, May 1989.
- [9] E. W. Coiera. Learning qualitative models from example behaviours. In *Proc. Third Workshop on Qualitative Physics*, Stanford, August 1989.
- [10] H. de Jong and A. Rip. The computer revolution is science: steps towards the realization of computer-supported discovery environments. *Artificial Intelligence*, 91:225–256, 1996.
- [11] D. DeCoste. Dynamic across-time measurement interpretation. *Artificial Intelligence*, 51:273–341, 1991.
- [12] L. Dehaspe, H. Toivonen, and R. D. King. Finding frequent substructures in chemical compounds. In *The Fourth International Conference on Knowledge Discovery and Data Mining*, pages 30–36, Menlo Park, CA., 1998. AAAI Press.
- [13] S. Dzeroski. Discovering dynamics: from inductive logic programming to machine discovery. *Informatica*, 16(4):30–41, 1992.
- [14] S. Dzeroski and L. Todorovski. Discovering dynamics: from inductive logic programming to machine discovery. *J. Intell. Information Syst.*, 4:89–108, 1995.
- [15] P. A. Flach and A. C. Kakas. *Abduction and Induction: Essays on their relation and integration*. Kluwer Academic Publishers, 2000.
- [16] S. M. Garrett, G. M. Coghill, R. D. King, and A. Srinivasan. On learning qualitative models of qualitative and real-valued data. Technical Report UWA-DCS-01-037, University of Wales, Aberystwyth, August 2001. Submitted to *Artificial Intelligence Sept’ 2001*.
- [17] D. T. Hau and E. W. Coiera. Learning qualitative models of dynamic systems. *Machine Learning*, 26:177–211, 1993.
- [18] A. C. Kakas, R. A. Kowalski, and F. Toni. Abductive logic programming. *J. of Logic and Computation*, 2:719–770, 1992.
- [19] H. Kay, B. Rinner, and B. Kuipers. Semi-quantitative system identification. *Artificial Intelligence*, 119:103–140, 2000.
- [20] R. D. King and A. Srinivasan. The discovery of indicator variables for qsar using inductive logic programming. *Journal of Computer-Aided Molecular Design*, 11:571–580, 1997.

- [21] I. C. Kraan, B. L. Richards, and B. J. Kuipers. Automatic abduction of qualitative models. In *Proceedings of Qualitative Reasoning 1991 (QR'91)*, 1991.
- [22] B. Kuipers. Commonsense reasoning about causality: Deriving behavior from structure. *Artificial Intelligence*, 24:169–204, 1984.
- [23] B. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.
- [24] B. Kuipers. *Qualitative Reasoning*. MIT Press, 1994.
- [25] J. E. Laird, P.S. Rosenbloom, and A. Newell. Chunking in soar: The anatomy of a general learning mechanism. *Machine Learning*, 1:11–46, 1986.
- [26] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.
- [27] S. Muggleton. *Inductive Logic Programming*. Academic Press, London, 1992.
- [28] S. Muggleton and C. Feng. Efficient induction of logic programs. In *Proc. of the First Conf. on Algorithmic Learning Theory*. OHMSHA, Tokyo, 1990.
- [29] G. Plotkin. *Automatic Methods of Inductive Inference*. PhD thesis, Edinburgh University, 1971.
- [30] S. Ramachandran, R. J. Mooney, and B. J. Kuipers. Learning qualitative models for systems with multiple operating regions. In *Working Papers of the Eighth International Workshop on Qualitative Reasoning about Physical Systems (QR-94)*, Nara, Japan, 1994.
- [31] A. C. C. Say and S. Kuru. Qualitative system identification: deriving structure from behavior. *Artificial Intelligence*, 83:75–141, 1996.
- [32] T. E. Shoup. *A Practical Guide to Computer Methods for Engineers*. Prentice-Hall Inc., Englewood Cliffs, N. J. 07632, 1979.
- [33] A. Srinivasan. *Aleph web site* :. web.comlab.ox.ac.uk/oucl/research/areas/mach-learn/Aleph/aleph_toc.html, 2000.
- [34] A. Srinivasan and R. D. King. Feature construction with inductive logic programming: A study of quantitative predictions of biological activity aided by structural attributes. *Data Mining and Knowledge Discovery*, 3:37–57, 1999.
- [35] L. Todorovski, A. Srinivasan, J. Whiteley, and D. Gavaghan. Discovering the structure of partial differential equations from example behavior. In *Proceedings of the Seventeenth International Conference on Machine Learning*, San Francisco, 2000.
- [36] A. Yamamoto. Revising the logical foundations of inductive logic programming systems with ground reduced programs. *New Generation Computing*, 17:119–127, 1999.