Artificial Intelligence in Medicine, , 1–51 (1995)

Learning Qualitative Models from Physiological Signals

DAVID T. HAU dave@welchlink.welch.jhu.edu
School of Medicine, The Johns Hopkins University, Baltimore, MD 21205, U.S.A.

ENRICO W. COIERA ewc@hplb.hpl.hp.com
Hewlett-Packard Laboratories, Filton Road, Stoke Gifford, Bristol BS12 6QZ, U.K.

Editor:

Abstract. Physiological models represent a useful form of knowledge, but are both difficult and time consuming to construct by hand. Further, most physiological systems are incompletely understood. This article addresses these two issues with a system that learns qualitative models from physiological signals. The qualitative representation of models allows incomplete knowledge to be encapsulated, and is based on Kuipers' approach used in his QSIM algorithm. The learning algorithm allows automatic generation of such models, and is based on Coiera's GENMODEL algorithm. We first show that QSIM models are efficiently PAC learnable from positive examples only, and that GENMODEL is an algorithm for efficiently constructing a QSIM model consistent with a given set of examples, if one exists. We then describe the learning system in detail, including the front-end processing and segmenting stages that transform a signal into a set of qualitative states, and GENMODEL that uses these qualitative states as positive examples to learn a QSIM model. Next we report results of experiments using the learning system on data segments obtained from six patients during cardiac bypass surgery. Useful model constraints were obtained, representing both normal physiology and knowledge particular to the pathophysiology of the patient being monitored. Model variation across time and across different levels of temporal abstraction and fault tolerance is examined.

Keywords: Patient monitoring, qualitative reasoning, machine learning, inductive logic programming, PAC learning, signal processing, cardiovascular system, physiologic modeling

1. Introduction

Physiological models have a central role in medical knowledge, encapsulating our understanding of experimentally observed physical processes. These encapsulations act both as theories whose predictions can be used for further research, and as clinical models whose predictions assist in delivery of therapy. For computer-based decision support systems, physiological models represent a useful form of knowledge because they encapsulate structural information of the system and allow deep forms of reasoning techniques to be applied. For example, such models are used in many prototype intelligent monitoring systems and medical expert systems. However, constructing physiological models by hand is difficult and time consuming. Further, most physiological systems are incompletely understood. These factors have hindered the development of model-based reasoning systems for clinical decision support.

Qualitative models permit useful representations of a system to be developed in the absence of extensive knowledge of the system. In the medical domain, such models have been applied to:

- diagnostic patient monitoring of acid-base balance [6]
- qualitative simulation of the iron metabolism mechanism [15]
- qualitative simulation of urea extraction during dialysis [2]
- qualitative simulation of the water balance mechanism and its disorders [18]
- qualitative simulation of the mechanism for regulation of blood pressure [18]

Recent developments in machine learning have produced methods of automatically inducing qualitative models from system behaviors. Applying such techniques to learning physiological models should not only benefit knowledge acquisition, but also provide a useful tool for physiologists who need to process vast amounts of data and induce useful theoretical models of the systems they study. The learning system could also serve as a tool for model-based diagnosis. For example, it could be incorporated into a diagnostic patient monitoring system to perform adaptive model construction for diagnosis in a dynamic environment.

The goal of this work is to determine whether useful physiological models can be generated by machine learning methods. Since these have up until now been applied to artificial data sets [4], [35], [27], we are particularly interested to see whether they scale up to real data sets, and to understand what signal processing techniques need to be used in addition to the learning mechanisms. We describe a system for learning qualitative models from physiological signals. The qualitative representation of physiological models used is based on Kuipers' approach, used in his qualitative simulation system QSIM [19]. The learning algorithm adopted is based on Coiera's GENMODEL system described in [5].

In our system, we use signals derived from hemodynamic measurements, including stroke volume, cardiac output, heart rate, arterial blood pressure and central venous pressure.

2. Qualitative Reasoning

In studying the behavior of dynamic systems, we often model the system with a set of differential equations. The differential equations capture the structure of the system by specifying the relationships that exist among the functions of the system. From the differential equations and an initial state, we can derive a quantitative system behavior using analytical methods or numerical simulation.

A qualitative abstraction of the above procedure allows us to work with an incomplete specification of the system. A qualitative model can be represented by a set of qualitative differential equations, or qualitative constraints. From the constraints

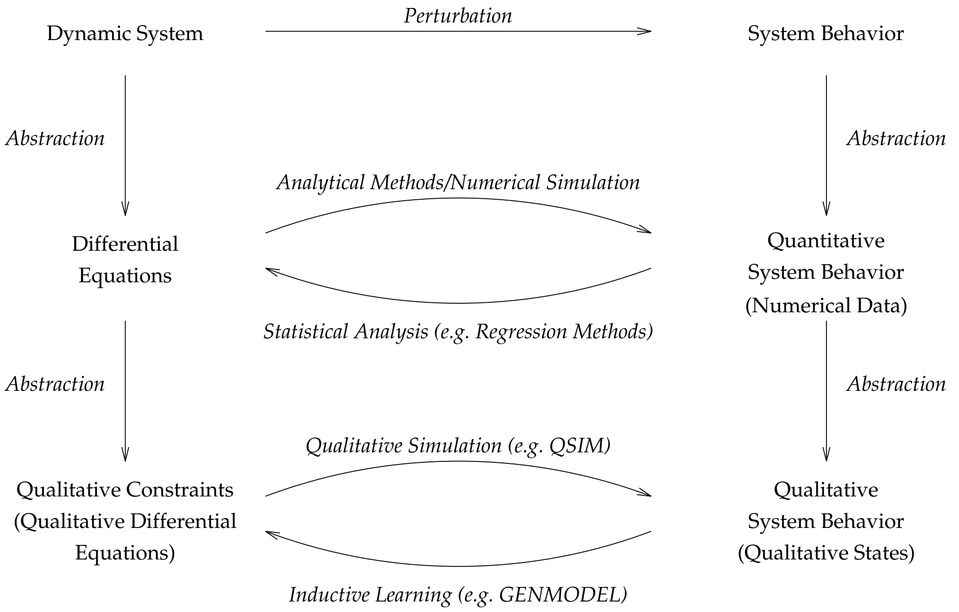

Figure 1. Qualitative reasoning is an abstraction of mathematical reasoning with differential equations and continuously differentiable functions.

and an initial state, we can derive a qualitative system behavior using qualitative simulation. Figure 1 illustrates this abstraction.

Different qualitative representations for models and their behavior have resulted from research in different problem domains [7]. In this section, we describe Kuipers' representation used in his qualitative simulation system QSIM [19].

2.1. Qualitative Model Constraints

QSIM represents a model of a system by a set of qualitative constraints applied to the functions of the system. These include arithmetic constraints, which correspond to the basic arithmetic and differential operators, and monotonic function constraints, which correspond to monotonically increasing and decreasing functions that exist between two functions.

2.1.1. Arithmetic Constraints

Four arithmetic constraints are included in the QSIM representation:

1. $add(f, g, h) \Longleftrightarrow f(t) + g(t) = h(t)$

2. $mult(f, g, h) \Longleftrightarrow f(t) \times g(t) = h(t)$

3. $minus(f, g) \Longleftrightarrow f(t) = -g(t)$

4. $deriv(f, g) \Longleftrightarrow f'(t) = g(t)$

2.1.2. Monotonic Function Constraints

When working with incomplete knowledge of a system, we may need to express a functional relationship that exists between two system functions, without specifying the functional relationship completely. In QSIM, the relationship can be described in terms of regions that are monotonically increasing or decreasing, and landmark values that the functions pass through. The QSIM representation includes two constraints for strictly monotonically increasing and decreasing functional relationships.

1. $M^+(f, g) \Longleftrightarrow f(t) = H(g(t))$ where $H(x)$ is a strictly monotonically increasing function

2. $M^-(f, g) \Longleftrightarrow f(t) = H(g(t))$ where $H(x)$ is a strictly monotonically decreasing function

Note that the two function constraints M^+ and M^- map onto many different functions including exponential, logarithmic, linear and other monotonically increasing or decreasing functions. This many-to-one mapping enables qualitative models to capture incomplete knowledge of a system.

Qualitative constraints can be considered as qualitative differential equations (QDE), representing an abstraction of ordinary differential equations (ODE). Every suitable ODE can be decomposed into a corresponding set of qualitative constraints.

2.2. Qualitative System Behavior

A dynamic system is characterized by a number of system functions which vary over time. The system behavior can be described in terms of these functions. In QSIM, system functions must be *reasonable functions* $f : [a, b] \rightarrow \Re^*$ where $\Re^* = [-\infty, \infty]$, which satisfy the following criteria:

1. f is continuous on $[a, b]$

2. f is continuously differentiable on (a, b)

3. f has only finitely many critical points in any bounded interval

4. $\lim_{t \rightarrow a^+} f'(t)$ and $\lim_{t \rightarrow b^-} f'(t)$ exist in $\Re^*$. We define $f'(a)$ and $f'(b)$ to be equal to these limits.

Every system function $f(t)$ has associated with it a finite and totally ordered set of *landmark values*. These include 0, the value of $f(t)$ at each of its critical points, and the value of $f(t)$ at each of the endpoints of its domain. The set of landmark values for a function form its quantity space which includes all the values of interest for the function. A value can be either at a landmark value, or in an interval between two landmark values.

The system has associated with it a finite and totally ordered set of *distinguished time points*. These include all time points at which any of the system functions reaches a landmark value. The set of distinguished time points form a time space. A qualitative time can be either a distinguished time point, or an interval between two adjacent distinguished time points.

The *qualitative state* of f at t is defined as a pair $< qval, qdir >$. $qval$ is the value of the function at t, and is either a landmark value or an interval between two landmark values. $qdir$ is the direction of change of the function at t, and is one of *inc*, *std*, or *dec* depending on whether the function is increasing, steady or decreasing at t respectively.

Since f is a reasonable function, the qualitative state of f must be constant over intervals between adjacent distinguished time points. Therefore a function can be completely characterized by its qualitative states at all its distinguished time points and at all intervals between adjacent distinguished time points. Such a temporal sequence of qualitative states of f form the *qualitative state history* or *qualitative behavior* of f.

Since a reasonable function f is continuously differentiable, the Intermediate Value Theorem and the Mean Value Theorem restrict the possible transitions from one qualitative state to the next. [19] includes a table listing all possible transitions.

2.3. Qualitative Simulation: QSIM

QSIM takes a qualitative model and an initial state, and generates all possible behaviors from the initial state consistent with the qualitative constraints in the model. Starting with the initial state, the QSIM algorithm works by repeatedly taking an active state and generating all possible next-state transitions according to the table of possible transitions mentioned in the previous section. These transitions are then filtered according to restrictions posed by the constraints in the system model. Because a model may allow multiple behaviors, QSIM builds a tree of states representing all possible behaviors. Any path across the tree from the given initial state to a final state is a possible behavior of the system.

2.4. Learning Qualitative Models: GENMODEL

GENMODEL [5] goes in the opposite direction to QSIM. It takes a system behavior and dimensional information about the system functions, and generates all qualitative constraints that permits the system behavior. The GENMODEL algorithm works by first generating all possible dimensionally correct qualitative constraints that may exist among the system functions, according to different permutations of the functions. Then it progresses along the state history, successively pruning all constraints that are inconsistent with each state transition. The set of qualitative constraints remaining at the end represent the *most specific* model that permits the given behavior. Any subset of this model also permits the given behavior, and therefore is also a possible model of the system.

3. Learning Qualitative Models

In this section, we examine the learnability of qualitative models employing the QSIM formalism [19]. The section starts with a detailed description of the GENMODEL algorithm [5], including the newly added features of dimensional analysis and fault tolerance. We then show that QSIM models are efficiently learnable under the Probably Approximately Correct (PAC) model of learning [34]. The proof is based on GENMODEL which can efficiently construct a QSIM model consistent with a given set of examples, if one exists. Next, we examine the difficulties of applying PAC results to our task of learning QSIM models from physiological signals. The section ends with a comparison of GENMODEL with other approaches of learning qualitative models.

3.1. GENMODEL

GENMODEL is a system that learns qualitative models by taking a sequence of states from a dynamic behavior and performing a relative least general generalization (RLGG) [25] on them [5]. [1] The algorithm works in a specific to general manner, and so the models that are learned are the most specific ones that explain the observed behaviors, and are expressed in the more general QDE concept language [8]. In this work, the original implementation of the GENMODEL program described in [5] is extended to include dimensional analysis and fault tolerance.

In brief, GENMODEL is presented with a sequence of qualitative state descriptions, describing a single time varying behavior. The program is equipped with background knowledge of legal QDE types, and rules for deciding which value combinations satisfy those qualitative equation types. GENMODEL takes the values of the time varying functions that compose a state description, and using the QDE definitions in its background knowledge as templates, generates an exhaustive set of possible QDEs. Each such QDE is a ground clause, instantiated with the parameter values presented in the example state, and together these represent all the combinations of QDE types and function pairings that may potentially exist. For example, given a state

$S : f(a) =< 0, inc >, f(b) =< -3, dec >, f(c) =< 3, std >$

and the QDE $add(x, y, z)$ which defines addition, GENMODEL would initially generate:

$add(a/0, b/-3, c/3)$
$add(a/0, c/3, b/-3)$
$add(b/-3, a/0, c/3)$
$add(b/-3, c/3, a/0)$
$add(c/3, a/0, b/-3)$
$add(c/3, b/-3, a/0)$
Each such ground clause is then tested against the mathematical QDE definitions in the background, and those that describe illegal value combinations are filtered. Successive states repeat the filtering process by testing the surviving clauses, and only those that match the behaviors encapsulated in the subsequent examples are retained. The result of the filtering process on the example state would be that GENMODEL retains:
$add(b/-3, c/3, a/0)$
$add(c/3, b/-3, a/0)$
since these satisfy the definition of *add*, and these clauses would be generalized to:
$add(b, c, a)$
$add(c, b, a)$
Since these are mathematically identical, one of the two clauses would be deleted by a straightforward redundancy check.

3.1.1. The Algorithm

Input:

- A set of system functions, $Functions$.
- A set of units for the system functions, $Units$
- A set of landmark lists for the system functions, $Landmarks$.
- A set of qualitative states, $States$.

Output:
A qualitative model which consists of all constraints that are consistent with the state history and dimensionally correct, $Model$.

Functions used:

$search()$ A function for searching corresponding values from a set of qualitative states.

$dimcheck()$ A function for checking dimensional compatibility of functions within a constraint.

$check()$ A function for checking validity of a constraint given a qualitative state and sets of corresponding values.

```
begin
    Constraints ← ∅;
    Correspondings ← search(States);
    for each f1, f2 in Functions such that f1 ≠ f2 do
        for each predicate2 in {inv, deriv, inv_deriv, M+, M−} do
            if dimcheck(predicate2, f1, f2, Units) then
                add predicate2(f1,f2) to Constraints;
    for each f1, f2, f3 in Functions such that f1 ≠ f2≠f3 do
        for each predicate3 in {add, mult} do
            if dimcheck(predicate3, f1, f2, f3, Units) then
                add predicate3(f1,f2,f3) to Constraints;
    for each s in States do
            for each c in Constraints do
                if not check(c, s, Landmarks, Correspondings) then
                    delete c from Constraints;
    reduce(Constraints);
    Model ← Constraints;
end
```

Figure 2. GENMODEL algorithm.

reduce() A function for removing redundancy from constraints. For example, since $M^+(A,B)$ and $M^+(B,A)$ specify the same relationship, one of them can be removed.

Method:

- Search the entire state history *States* for sets of corresponding values.
- Generate the initial search space by constructing all dimensionally correct constraints with different permutations of system functions in *Functions*.
- Successively prune inconsistent constraints upon each qualitative state in *States*.
- Remove redundancy from the remaining constraints
- Output the result as a qualitative model.

The entire algorithm is shown in Figure 2.

3.1.2. Dimensional Analysis

Dimensional analysis is an effective way of significantly reducing the size of the initial search space of constraints. Before generating a constraint, GENMODEL checks for units compatibility among functions within the constraint. This is similar to the approach used in several other inductive learning systems, including ABACUS [10], a system for quantitative discovery, and MISQ [27], a system based upon GENMODEL.

The dimension of each function is specified at the beginning, usually in terms of the type of quantity the function represents, e.g. $1/time$ for the heart rate (HR), $volume$ for the stroke volume (SV), and $volume/time$ for the cardiac output (CO). This allows the constraint $mult(HR, SV, CO)$ to be generated since $(1/time) \times (volume) = volume/time$, but does not allow $mult(HR, CO, SV)$ or $add(HR, SV, CO)$ to be generated since they are dimensionally incorrect. Note that the functional constraints M^+ and M^- are not restricted by dimensions.

3.1.3. Performance on Learning the U-Tube Model

With dimensional analysis, GENMODEL comes up with exactly the six constraints for the U-tube system, given the three qualitative states describing the example behavior. The U-tube system is a standard reference problem in the qualitative modeling community [4], [5]. This is a significant improvement to previous results reported in [5] in which 14 constraints were obtained.

3.1.4. Fault Tolerance

For domains involving noisy learning data, such as hemodynamic monitoring, it is difficult to implement front-end signal processing which filters the noise and restores the signals completely. Part of the difficulty lies in noise detection. Hemodynamic monitoring is vulnerable to a wide variety of artifacts. These include artifacts which are easy to detect because their values are outside the physiologically attainable range, as well as ones which are hard to detect because the signals are not affected as drastically. For example, artifacts caused by various clinical interventions are usually easy to detect, while those caused by mild patient movements can be hard to detect. Even if artifacts are detected, it is hard if not impossible to restore a signal heavily distorted by artifacts. To accommodate such difficulties in obtaining a perfectly clean signal for segmentation, we need to incorporate a degree of fault tolerance into GENMODEL.

A simple approach is to tag a counter onto every constraint in the initial search space. This counter keeps track of how many states the constraint has failed in so far. We set a noise level η to a fraction of the total number of states in the example behavior. A constraint has to be inconsistent with this many states before it is pruned.

3.2. Probably Approximately Correct Learning

A common setting in machine learning is as follows: Given a set of examples, produce a concept consistent with the examples that is likely to correctly classify future instances. We are interested in algorithms that can perform this task efficiently. The Probably Approximately Correct (PAC) model of learning introduced by Valiant [34] is an attempt to make precise the notion of "learnable from examples" in such a setting. [16] and [28] describe this model in detail.

3.2.1. PAC Learnability

Stated informally, PAC learnability is the notion that the concept acquired by the learner should closely approximate the concept being taught, in the sense that the acquired concept should perform well on new data drawn according to the same probability distribution the examples used for learning are drawn.

To define PAC learnability formally, we say that a concept class C is efficiently PAC learnable if there exists an algorithm A and a polynomial $s(\cdot\,,\cdot\,,\cdot)$ such that for all n, ϵ, and δ, all probability distributions P_n on X_n, and all concepts $c \in C_n$, A will with probability at least $1-\delta$, when given a set of examples of size $m = s(n, \frac{1}{\epsilon}, \frac{1}{\delta})$ drawn according to P_n, output a $c' \in C_n$ such that $error(c') \leq \epsilon$. Furthermore, A's running time is polynomially bounded in n and m.

3.2.2. Proving PAC Learnability

One approach of PAC learning due to Blumer *et al.* [3] is as follows: Draw a large enough set of examples according to P_n, and find an algorithm which, given the set of examples, outputs *any* concept $c \in C_n$ consistent with all the examples in polynomial time. If there exists such an algorithm for the concept class C, C is said to be *polynomial-time identifiable.*

Next we look at how large a set of examples is "large enough" for PAC learning. [3] shows that a sample size m satisfying the following lower bound is sufficient: [2]

$$m = \Omega\,(\frac{1}{\epsilon}\,(ln\,|C_n| + ln\,\frac{1}{\delta}))$$

C_n is polynomial-sized if $ln\,|C_n|$ is polynomial in n. Therefore if C_n is polynomial-sized, then m is polynomial in n, $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$.

An algorithm that draws at least this many examples according to P_n and outputs any concept consistent with all the examples in polynomial time is a PAC learning algorithm. Thus if C_n is polynomial-sized and polynomial-time identifiable, then it is efficiently PAC learnable.

3.2.3. An Occam Algorithm for Learning Conjunctions

In [34] Valiant provides an algorithm for PAC learning the set of boolean formulae in conjunctive normal form (CNF) where each clause contains at most k literals. This set of boolean formulae is known as *k-CNF*. The algorithm is capable of PAC learning from positive examples only. In Section 3.2.4 we will map the problem of identifying a QSIM model consistent with a given set of examples to the problem of identifying a *k-CNF* consistent with a given set of examples.

First we calculate the number of examples needed. The number of conjunctions over the boolean variables $x_1, \ldots, x_n$ is 3^n since each variable either appears as a positive or negative literal, or is absent entirely. Applying the formula for the lower bound in the previous section, we see that a sample of size $O(\frac{n}{\epsilon} + \frac{\ln 1/\delta}{\epsilon})$ is sufficient to guarantee that the hypothesis output by our learning algorithm is ϵ accurate with confidence of at least $1 - \delta$.

The algorithm starts with the hypothesis conjunction which contains all the literals:

$$c' = x_1 \wedge \bar{x}_1 \wedge \cdots \wedge x_n \wedge \bar{x}_n$$

Upon each positive example x, the algorithm updates c' by deleting the literal x_i if $x_i = 0$ in the example, and deleting the literal $\bar{x}_i$ if $x_i = 1$ in the example. Thus the algorithm deletes any literal that contradicts the data. This can be seen as starting with the most specific concept and successively generalizing the concept upon each positive example given.

Since the algorithm takes linear time to process each example, given m examples with m as calculated above, the running time is bounded by mn and hence is bounded by a polynomial in n, $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$. Therefore this is an efficient PAC learning algorithm for the class of *k-CNF*.

3.2.4. QSIM Models are PAC Learnable

From Section 3.2.2 we conclude that to prove that the concept class of QSIM models is PAC learnable, it suffices to prove that the class is polynomial-sized and that it is polynomial-time identifiable. The following two sections provide these proofs.

The Class of QSIM Models is Polynomial-Sized

To show that the concept class of QSIM models is polynomial-sized, we begin by noting that in the QSIM formalism there are five kinds of two-function constraints (*inv*, *deriv*, *inv_deriv*, M^+ and M^-), and two kinds of three-function constraints (*add* and *mult*). Therefore with n system functions, the number of possible QSIM constraints N is as follows:

$$N = 5n(n-1) + 2n(n-1)(n-2)$$

Therefore, the number of possible QSIM models is:

$$|QSIM\text{–}Models(n)| = 2^N = 2^{5n(n-1)+2n(n-1)(n-2)}$$

since each QSIM constraint can either be present or absent in the model. This implies that:

$$\lg(|QSIM\text{–}Models(n)|) = N = O(n^3)$$

Therefore the concept class of QSIM models is polynomial-sized.

To PAC learn a QSIM model, we need m examples where m is calculated as follows:

$$m = \Omega\left(\frac{1}{\epsilon}\left((5n(n-1) + 2n(n-1)(n-2))\ln 2 + \ln\frac{1}{\delta}\right)\right)$$

In practice, the initial concept space can be constrained to a much smaller size by removing redundant and dimensionally incorrect constraints prior to learning. This substantially reduces the sample size required for learning (see Section 3.2.5).

QSIM Models are Polynomial-Time Identifiable

In this section we show that GENMODEL is an algorithm for efficiently constructing a QSIM model consistent with a given set of examples. We prove this by mapping the problem of identifying a QSIM model consistent with a given set of examples to the problem of identifying a *k-CNF* consistent with a given set of examples. The algorithm presented in Section 3.2.3 then yields an algorithm for identifying a QSIM model consistent with a given set of examples in polynomial time. We show that this algorithm is in fact identical to GENMODEL.

We view each QSIM model as a conjunction of QSIM constraints, and each QSIM constraint as a boolean variable. Then learning QSIM models is equivalent to learning monotone conjunctions [3] with N boolean variables, where N is the number of possible QSIM constraints as calculated in the previous section.

The algorithm starts with the hypothesis of a monotone conjunction which contains all N of the boolean variables, i.e. all possible QSIM constraints:

$$c' = x_1 \wedge \cdots \wedge x_N$$

The qualitative states provided for learning constitute the positive examples. Upon each positive example x, the algorithm updates c' by deleting the boolean variable x_i if the corresponding QSIM constraint is inconsistent with the example. Since each boolean variable x_i corresponds to a QSIM constraint, the algorithm prunes any constraint that is inconsistent with each qualitative state. This can be seen as starting with the most specific model and successively generalizing the model upon each qualitative state given. This is identical to the approach taken by GENMODEL.

Now it remains to show that GENMODEL takes polynomial time to process each qualitative state. We review each step taken by GENMODEL in learning a QSIM model:

- Search the entire state history for sets of corresponding values. For m qualitative states, there are at most m sets of corresponding values, and the search takes $O(m)$ time.
- Generate the initial search space by constructing all dimensionally correct constraints with different permutations of system functions. For n system functions, this takes $O(n^3)$ time as shown in the previous section.
- Successively prune inconsistent constraints upon each qualitative state. Checking for consistency of a constraint with a qualitative state involves:
  - Checking landmark values and directions of change. This takes linear time.
  - Checking corresponding values. Since there are at most m sets of corresponding values, this takes $O(m)$ time.
- Remove redundancy from the remaining constraints. Since we started off with $O(n^3)$ constraints, there are at most the same number of constraints remaining in the final model. Therefore, removing redundancy takes $O(n^3)$ time.
- Output the result as a qualitative model.

Therefore for GENMODEL the processing time of each qualitative state is polynomial in m and n.

Since the algorithm takes polynomial time to process each qualitative state, given m states with m as calculated above, the running time is bounded by $p(m, n)$ where $p(\cdot\,,\cdot)$ is a polynomial in the two arguments. Therefore the running time is bounded by a polynomial in n, $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$. Therefore this is an efficient PAC learning algorithm for the concept class of QSIM models.

3.2.5. Applicability of PAC Learning

As discussed in Section 3.2.4, the following is a PAC learning algorithm for learning a QSIM model from physiological signals:

1. Obtain m qualitative states where m is calculated as follows:

$$m = \Omega\,(\frac{1}{\epsilon}\,(\ln 2^N + \ln \frac{1}{\delta}))$$

N is the number of possible QSIM constraints which are non-redundant and dimensionally correct. For our experiments, we use 8 different signals. Out of the 952 different permutations, only 99 of them represent non-redundant and dimensionally correct constraints. Therefore $N = 99$.

2. Learn a QSIM model from the m qualitative states using GENMODEL.

Applying this algorithm to our learning task is difficult for the following reasons:

- Qualitative states cannot be modeled as *independent* examples drawn from an underlying probability distribution. Given a reasonable function and a qualitative state, there are only a limited number of possible transitions the system can make, as described in [19]. Further, successive states in signals obtained from hemodynamic monitoring are highly correlated because of physiological constraints limiting for instance the rate of change of signals.

- For our experiments, $N = 99$. To PAC learn a QSIM model with an accuracy and a confidence level of 80%, i.e. $\epsilon = \delta = 0.2$, we need $m = 352$ qualitative states. To do so with an accuracy and a confidence level of 60%, i.e. $\epsilon = \delta = 0.4$, we need $m = 174$ qualitative states. In Sections 5.2.3 and 5.2.4, we show that the standard deviation parameter σ of the Gaussian filter represents the level of temporal abstraction for learning. Smaller values of σ correspond to modeling of faster processes and larger values correspond to modeling of slower processes. For a given length of patient data, a Gaussian filter with a smaller σ produces less smoothing of the signals, and therefore more qualitative states. Therefore, the above sample sizes are more easily achieved with a smaller σ. This is reasonable since learning slower processes requires observing the patient for a longer period of time, and vice versa. For learning slow processes, the above sample sizes may not be achievable at all since they require such a long time span that the patient condition and therefore the corresponding model may change during the modeling period.

- Signals from hemodynamic monitoring are corrupted by various artifacts and noise. The PAC learning algorithm previously developed assumes learning examples to be noise-free.

Because of the above difficulties of applying the PAC model to our learning task, we will not follow the PAC learning algorithm strictly. We will apply GENMODEL for polynomial-time identification of a QSIM model from qualitative states only. Even so, as we will see in Section 6, we still obtain useful models of reasonable size.

3.3. Comparison of GENMODEL with Other Learning Approaches

The ability to generate ground clauses using only positive examples and non-ground background rules distinguishes GENMODEL from other first order inductive learning systems like FOIL [26] or GOLEM [22], which seem unsuited to the learning of qualitative models, since they require all knowledge to be expressed in ground clauses. Further, GOLEM is unable to learn non-deterministic concepts like *add* [4]. Given a complete and correct set of qualitative behaviors, GENMODEL will learn a set of QDEs which is the most specific set that reproduces those behaviors. The following sections describe in detail the differences between GENMODEL and other approaches of learning qualitative models.

3.3.1. GENMODEL does not require negative examples.

The greatest strength of GENMODEL is that it learns from positive examples only. There is no need to generate negative examples as needed in other inductive learning approaches such as GOLEM and genetic algorithms.

In [4], Bratko et al. report that learning the U-tube model with GOLEM requires six hand-generated negative example states, in addition to the same positive example behavior we used for GENMODEL which consists of only three states. On each iteration in the GOLEM algorithm, a fixed number of clauses are first generated by relative least general generalization (RLGG) [25]. The clause that covers the most positive examples and none of the negative examples is then chosen for propagation to the next iteration.

In [35], Varšek's genetic algorithm approach requires 17 positive example states and 78 negative example states to learn the U-tube model. In each cycle, candidate solutions are selected for "reproduction" based on a fitness function which is the sum of the fraction of positive and negative examples covered correctly and a "bonus" indicating the size of the solution.

In both approaches, it is essential for the user to give the "right" negative examples. Badly chosen negative examples or an inadequate number of them will cause an inappropriate clause to be propagated to the next iteration, which will ultimately affect the concept output in the end. However, there are no existing rules to guide the selection of negative examples. A trial-and-error approach can be tedious, especially in complex domains such as human physiology.

3.3.2. GENMODEL does not require ground facts for background knowledge

In GENMODEL, the definitions of the QSIM representation are inherent in the *check*() function used for checking consistency of a constraint with a given qualitative state. There is no need to generate explicit ground facts [4] for this background knowledge, as needed in GOLEM.

GOLEM accepts definitions of background predicates in terms of ground facts. In learning QSIM models, explicit ground facts describing QSIM constraint definitions must be generated according to functions and landmark lists relevant to the modeling problem at hand. In [4], Bratko et al. report that learning the U-tube model requires a total of 5408 ground facts as background knowledge. This is already a simplification which excludes rules regarding corresponding values in the

M^+ and M^- constraints, rules regarding consistency of infinite values in the *add* constraint, and rules on the *mult* constraint. In a more complex domain such as human physiology which potentially involves long landmark lists, the size of the background knowledge required can grow exceedingly large.

3.3.3. GENMODEL is guaranteed to produce a correct model if one exists.

Given a set of qualitative states representing a system behavior, GENMODEL successively prunes inconsistent constraints upon each state. The constraints remaining in the end forms the output model. Therefore, GENMODEL is guaranteed to produce a correct model if one exists.

On the other hand, GOLEM and genetic algorithms perform heuristic searches across the concept space. GOLEM performs hill climbing with positive and negative example coverage as the heuristic guiding the search. Genetic algorithms similarly perform hill climbing with the fitness function serving as the heuristic. Since neither heuristic is a perfect quality measurement of the current model, GOLEM and genetic algorithms are not guaranteed to produce a correct model even if one exists, unless the search becomes exhaustive.

4. Physiological Signals and Models

4.1. Hemodynamic Monitoring

Hemodynamic monitoring provides information on the performance of the cardiovascular system (CVS), and covers many aspects of the CVS, including heart rate, blood pressures, temperatures, oxygen supply and others. In this work, we use eight signals derived from such measurements for learning a qualitative model that describes the CVS. The basic measurements and their characteristics will be reviewed in the following section.

4.2. Primary Measurements

4.2.1. Heart Rate (HR)

The heart rate is determined from the electrocardiogram (ECG) signal as the reciprocal of the interval between two successive QRS complexes. (QRS complexes are the large voltage spikes that correspond to ventricular contraction.) In our experiments, the heart rate signal is sampled at 1 Hz.

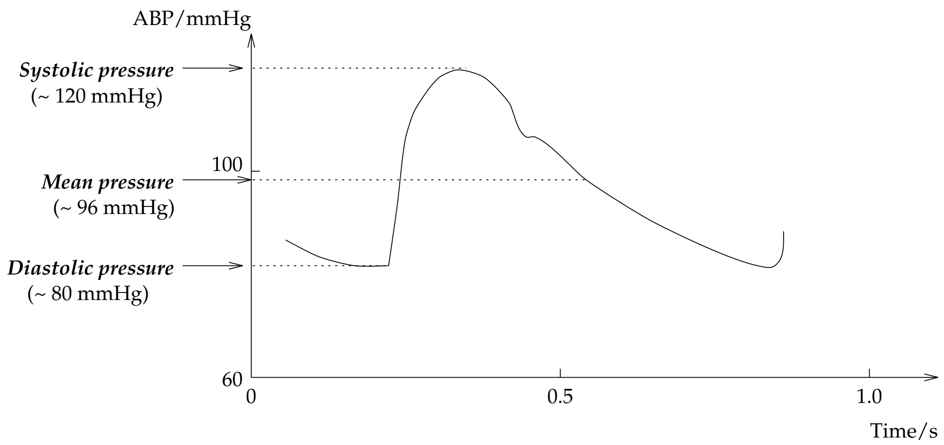

Figure 3. Deriving the systolic, diastolic and mean pressures from the arterial blood pressure waveform.

4.2.2. Arterial Blood Pressure Waveform (ABP)

The arterial blood pressure waveform in our data comes from invasive monitoring by a catheter. The signal in our data is sampled at 125 Hz. From the ABP waveform, the following signals are derived at 1 Hz (Figure 3):

Systolic arterial blood pressure (ABPS) - the value at the height of an ABP pulse.

Diastolic arterial blood pressure (ABPD) - the value at the lowest point of an ABP pulse.

Mean arterial blood pressure (ABPM) - the mean value of an ABP pulse. This can be calculated by dividing the area under the pulse by the duration of the pulse:

$$ABPM = \frac{\int_{t_1}^{t_2} ABP\,dt}{t_2 - t_1}$$

where t_1 is the starting time of the pulse and t_2 is the ending time of the pulse [12].

4.2.3. Central Venous Pressure (CVP)

Blood from all the systemic veins flows into the right atrium. Therefore, the pressure in the right atrium is called the central venous pressure. The central venous pressure in our data comes from invasive monitoring by a catheter. The waveform is sampled at 125 Hz. The mean CVP signal (CVPM) at 1 Hz is used in our experiments.

4.2.4. Temperature (T_{skin}, T_{core})

The skin temperature and the core temperature are recorded at 1 Hz. From these two signals, a skin-to-core temperature gradient can be determined as described in Section 4.3.4.

4.3. Derived Values

From the primary measurements, various useful indices of cardiovascular function may be calculated as follows [23], [29].

4.3.1. Stroke Volume (SV)

The difference between the end-diastolic volume and the end-systolic volume is the volume of blood pumped out of the ventricles during systole. This volume is about 70 ml, and is called the *stroke volume* [12].

We use the pulse contour method developed by Wesseling *et al.* [37]. This method estimates the stroke volume from the arterial blood pressure waveform as the systolic ejection area A_{sys} (Figure 4) divided by a constant Z_{ao} representing the characteristic impedance of the aorta:

$$SV = \frac{A_{sys}}{Z_{ao}} = \frac{\int_{t \in SEP}(ABP - ABPD)\,dt}{Z_{ao}}$$

Z_{ao} has been shown to be relatively constant over short periods. Therefore the systolic ejection area alone reflects the *qualitative* behavior of the function.

4.3.2. Cardiac Output (CO)

The cardiac output is the rate of blood flow from the left ventricle into the aorta. It is related to the heart rate and the stroke volume by the following equation:

$$CO = HR \times SV$$

4.3.3. Ventricular Contractility (VC)

$$VC = \left.\frac{d(ABP)}{dt}\right|_{peak}$$

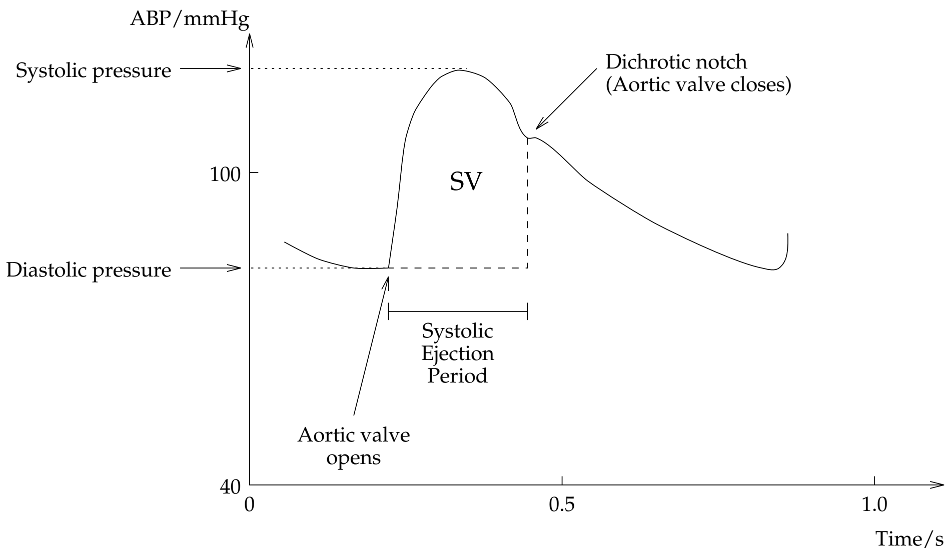

Figure 4. Deriving the stroke volume from the arterial blood pressure waveform.

Experimental studies have shown that the rate of rise of arterial blood pressure in general correlates well with the strength of contraction of the ventricle. The peak $\frac{d(ABP)}{dt}$, which occurs at the onset of systole, is often used as an indicator of ventricular contractility [29].

4.3.4. Skin-to-core Temperature Gradient (ΔT)

$$\Delta T = T_{core} - T_{skin}$$

The difference between the skin and core temperatures of the body is a good indicator of the degree of vasoconstriction. A rise in this differential indicates increasing vasoconstriction while a fall indicates vasodilation. The degree of vasoconstriction in turn reflects cardiac output. Under conditions of poor cardiac output, as in hypovolemia, the body responds by trying to raise the blood pressure by vasoconstriction, at the expense of tissue perfusion [23], [39].

4.3.5. Rate Pressure Product (RPP)

$$RPP = HR \times ABPS$$

The rate pressure product (RPP) is the product of the heart rate (HR) and the systolic arterial blood pressure (ABPS). Studies in animals [38] and humans [11] have shown that the RPP correlates well with the myocardial oxygen consumption (m$\dot{V}O_2$), which is closely related to the work of the heart. m$\dot{V}O_2$ depends on several factors, including heart rate, ventricular contractility and ventricular wall tension.

4.4. A Qualitative Cardiovascular Model

In this section, we describe a set of possible qualitative constraints that may exist among the different variables of hemodynamic monitoring described in the previous section. These constraints form a "gold-standard" target model of the CVS which allows us to compare our experimental results and evaluate the performance of the learning system.

Because of the enormous complexity of the CVS, formulating a model is by no means a simple task [33], [36]. The constraints included in this section are not meant to be a comprehensive coverage of the system. Also, different models may exist for different clinical conditions. A constraint may be valid only under certain conditions.

4.4.1. Relationship Among Heart Rate, Stroke Volume and Cardiac Output

The heart rate (HR), stroke volume (SV) and cardiac output (CO) are related by the following equation:

$$CO = HR \times SV$$

This translates into the following qualitative constraint:

$$mult(HR, SV, CO)$$

4.4.2. Relationship Among Heart Rate, Arterial Blood Pressure and Rate Pressure Product

The heart rate (HR), systolic arterial blood pressure (ABPS) and rate pressure product (RPP) are related by the following equation:

$$RPP = HR \times ABPS$$

Since the behavior of the mean arterial blood pressure (ABPM) approximates that of the systolic arterial blood pressure (ABPS) well in most circumstances, the following qualitative constraint is valid in general:

$$mult(HR, ABPM, RPP)$$

4.4.3. The Frank-Starling Law of the Heart

The Frank-Starling law states that within physiological limits, the heart pumps all the blood that comes to it without allowing excessive damming of blood in the veins. This translates into the following qualitative constraint:

$$M^{+}(CVPM, CO)$$

4.4.4. Heterometric & Homeometric Autoregulation of the Heart

When the cardiac muscle becomes stretched an extra amount, as it does when extra amounts of blood enter the heart chambers, the stretched muscle contracts with a greatly increased force, thereby automatically pumping the extra blood into the arteries. This ability of the heart to contract with increased force as its chambers are stretched is sometimes called *heterometric autoregulation of the heart.*

Further, when the heart is stretched, changes in heart metabolism cause an additional increase in contractile strength. It takes approximately 30 seconds for this effect to develop fully, an effect called *homeometric autoregulation* [12].

Therefore, within the physiological limit of the heart, the ventricular contractility (VC) of the heart increases with the stroke volume (SV):

$$M^{+}(SV, VC)$$

4.4.5. Effect of Heart Rate on Cardiac Output

An increase in heart rate can be caused by a higher oxygen demand in tissues and organs, as in physical exercise, or as a compensatory mechanism for a decreased arterial blood pressure, as in hypovolemia. This results in two different sets of constraints representing different conditions.

In normal condition, the more times the heart beats per minute, the more blood it can pump, since the stroke volume stays roughly the same. This can be seen from the equation

$$CO = HR \times SV$$

Therefore, the following qualitative constraint holds:

$$M^{+}(HR, CO)$$

Indeed, a rise in heart rate increases the net influx of calcium ions per minute into the myocardial cells, and enhances ventricular contractility:

$$M^{+}(HR, VC)$$

This increased contractility causes blood to be pumped out faster from the ventricle, shortens the systolic interval, allocates a larger proportion of the cardiac cycle to diastolic ventricular filling, and therefore maintains the stroke volume at a reasonable level.

However, once the heart rate exceeds a critical level (150-170 beats per minute in normal individuals) as in hypovolemia, the heart strength itself decreases, presumably because of overutilization of metabolic substrates in the cardiac muscle:

$$M^-(HR, VC)$$

This results in a significant decrease in diastolic filling time and consequently a decrease in the stroke volume:

$$M^-(HR, SV)$$

The tolerable limits for heart rate decrease with underlying cardiovascular impairment. For example, ventricular tachycardia in a patient with recent myocardial infarction produces a significant reduction in stroke volume, cardiac output, and subsequently arterial blood pressure also [12], [39].

4.4.6. Compensatory Mechanisms for Hypovolemia

Hypovolemia refers to an absolute and often sudden reduction in circulating blood volume relative to the capacity of the vascular system [30]. Activation of the sympathetic nervous system results in a series of compensatory mechanisms including:

- arteriolar vasoconstriction with resultant decreased perfusion to skin, skeletal muscle, kidney, and splanchnic organs. This causes an increase in the skin-to-core temperature gradient ΔT:

  $$M^-(CO, \Delta T)$$

- increased heart rate (tachycardia):

  $$M^-(CO, HR)$$

- increased myocardial contractility:

  $$M^-(CO, VC)$$

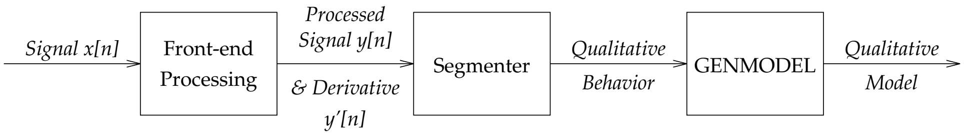

Figure 5. Overall architecture of the learning system.

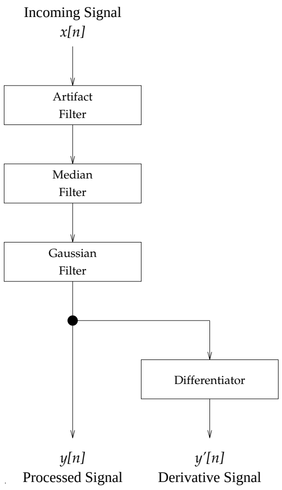

Figure 6. Architecture used for front-end processing of physiological signals.

5. System Architecture

5.1. Overview

The goal of the learning system is to generate qualitative models from physiological signals. The overall architecture of the system is illustrated in Figure 5.

The physiological signal is first processed by a front-end system, which outputs a filtered signal and its derivative. These are entered into the segmenter to produce a qualitative behavior in terms of a set of qualitative states. GENMODEL then uses this qualitative behavior to generate a qualitative model. [5] This section describes the front-end system and the segmenter in detail.

5.2. Front-End System

The architecture used for front-end processing of physiological signals is shown in Figure 6.

The signal first passes through an artifact filter which removes various artifacts and linearly interpolates the intervals of the artifacts removed. The resulting signal is then processed by a median filter which removes impulsive features lasting shorter than half the length of the filter window. A Gaussian filter then smooths the signal to the desired level of temporal abstraction by convolving it with a Gaussian kernel of an appropriate standard deviation σ. Finally, this smoothed signal is passed through a differentiator to obtain its derivative. The smoothed signal and its derivative are passed on to the segmenter for segmentation, producing a set of qualitative states describing the system behavior represented by the signal.

The Gaussian filter and the differentiator are implemented as finite impulse response (FIR) filters. For simplicity, the same length L is used for the median filter window and the impulse responses of the Gaussian filter and the differentiator. Also, since we are interested only in the *qualitative* behavior represented by the signals, and not in the absolute magnitudes, certain constant factors in the impulse responses have been omitted for simplicity.

5.2.1. Artifact Filter

The artifact filter removes artifacts caused by various clinical interventions, and linearly interpolates the intervals of the removed artifacts. The filter employs a simple threshold approach in artifact detection. When the magnitude of a signal rises above or falls below a certain threshold which cannot be physiologically attained in general, the filter removes the abnormal values and interpolates the interval with the last normal value. This approach may not work well for signals with complex artifacts or artifacts of long durations. For recognition of complex artifacts, a knowledge-based system [13] or a neural network [17] can be used.

5.2.2. Median Filter

The median filter operates by centering a window of a given length L at each point of the signal. The output at that point is the median of the values covered by the window. The median filter removes impulsive features in the signal with durations less than half of its window length, while retaining sharp edges of remaining features. (Unlike filters based on convolution, the median filter outputs only values existing in the input signal. It does not produce any new averaged values.) This property of the median filter makes it useful for removing short impulsive artifacts in the physiological signals prior to smoothing by the Gaussian filter. This prevents the impulsive artifacts from distorting the smoothed signals.

5.2.3. Temporal Abstraction

A complex system such as the human cardiovascular system involves processes operating at different time scales. From the same set of signals, depending on the particular time scale we are interested in, different sets of qualitative states and therefore different models can be obtained.

In [20] Kuipers describes a temporal abstraction relation among mechanisms operating at significantly different time scales. Processes that occur significantly faster than the time scale of a model can be considered as instantaneous in the model, while those that occur much slower can be considered as constant. For example, if we look at a system on the order of hours, processes that occur within seconds can be considered as instantaneous, while those occurring over days can be viewed as constant. Therefore if we perturb a system by increasing a function A, and observe that another function B responds to this change within seconds by increasing its value, then we can still model the relationship between A and B with the functional constraint $M^+(A, B)$ even though there is a delay between the perturbation and the response, since the response within seconds is seen as occurring instantaneously at this time scale.

We incorporate this idea of temporal abstraction into our learning system by the following scheme:

1. First, we use a Gaussian filter to remove changes lasting significantly shorter than the time scale we are interested in. This is described in Section 5.2.4.

2. Next, we implement the segmenter in such a way that critical points of different functions occurring within τ sampling periods are labelled as occurring at the same distinguished time point, where the parameter τ corresponds to the particular time scale at which we are interested in learning. This is described in Section 5.3.

Without the first step, there is a danger of *aliasing* in the temporal abstraction process. Features lasting for short durations ($< \tau$) can be aliased into one lasting for a long duration [14].

5.2.4. Gaussian Filter

The idea of using a Gaussian filter to analyze changes in a signal at different scales is borrowed from the well known technique of *scale-space filtering* in edge detection. Scale-space filtering constructs hierarchic symbolic signal descriptions by transforming the signal into a continuum of versions of the original signal convolved with a kernel containing a scale parameter. In an image, changes of intensity take place at many spatial scales depending on their physical origin. Marr and Hildreth [21] observed that detecting zero crossings in the Laplacian of the intensity values across different scales enables a system to distinguish between a physical edge from surface markings or shadows. They suggested that the original image be bandlimited at several different cutoff frequencies and that an edge detection algorithm be applied to each of the images. The resulting edge maps have edges corresponding to different scales.

In our learning system, we need to segment a set of signals at different time scales. We can do so by bandlimiting our original signals at several different cutoff frequencies and segmenting the signals by detecting zero crossings in the first derivative of the signals at different scales. The segmentation then produces a set of qualitative behaviors at different time scales which can be given to GENMODEL to produce qualitative models at different scales.

To bandlimit an image at different cutoff frequencies, the impulse response of the lowpass filter proposed by Marr and Hildreth is Gaussian shaped. This choice is motivated by the fact that the Gaussian function is smooth and localized in both the spatial and frequency domains. [6] A smooth impulse response is less likely to introduce any changes that are not present in the original shape. A localized impulse response is less likely to shift the location of edges. Further, Yuille and Poggio [40] and Babaud *et al.* [1] have separately shown that the Gaussian filter has a unique property concerning zero crossings of the first derivative of the filtered signal: [7] moving from coarse to fine scale, new zero crossings appear, but existing ones never disappear. Consequently, the extrema can be used to construct a tree describing the successive partitioning of the signal into finer subintervals as new zero crossings appear at finer scales. This partitioning of the signal by extrema moving from coarse to fine scale forms a strict hierarchy. Scale-space filtering in edge detection can be seen as a form of the more general technique of wavelet transforms in multiresolution signal analysis, with the wavelets here being Laplacians of shifted Gaussians, and signal edges located by zero crossings of the wavelet transform [32].

We adopt a similar approach for segmenting our signals. The impulse response of the lowpass filter used is based on the Gaussian function:

$$g(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{t^2}{2\sigma^2}}$$

for $-\infty < t < \infty$ and $\sigma > 0$. The standard deviation σ determines the cutoff frequency with a larger σ corresponding to a lower cutoff frequency. σ therefore determines the time scale we are operating at, with a smaller σ corresponding to a finer time scale and a larger σ corresponding to a coarser scale. The frequency response of the lowpass filter is the Fourier transform of $g(t)$ which is also Gaussian shaped:

$$G(\Omega) = e^{-\frac{\Omega^2\sigma^2}{2}}$$

To translate this into a discrete-time filter, we simply replace t with n, yielding $g[n]$ as follows:

$$g[n] = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{n^2}{2\sigma^2}}$$

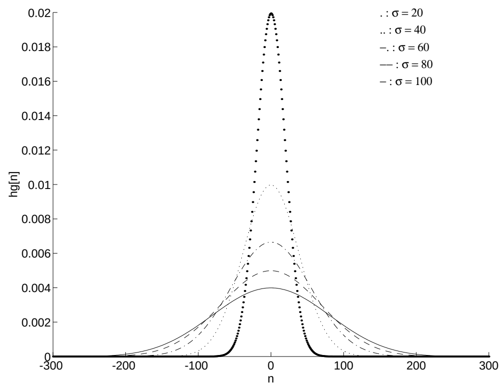

Figure 7. Plots of impulse responses $h_g[n]$ of Gaussian filters for $\sigma = 20, 40, 60, 80, 100$.

for $-\infty < n < \infty$ and $\sigma > 0$.

To obtain a finite impulse response (FIR) $h_g[n]$ from the infinite impulse response $g[n]$, we multiply $g[n]$ by the Hanning window $w[n]$:

$$h_g[n] = g[n]w[n]$$

In our experiments, we use values of σ at 10, 20, 40, 60, 80 and 100. $h_g[n]$ is plotted as shown in Figure 7 for these values of σ.

M is the order of the FIR filter. Therefore $M + 1$ is the length of the impulse response:

$$L = M + 1$$

In our learning system, M is set so that the finite impulse response extends to three standard deviations from the origin:

$$\frac{M}{2} = 3\sigma$$

which yields:

$$M = 6\sigma$$

Table 1 shows the lengths and the orders of the filters that correspond to the six values of σ we used in our experiments:

The frequency response $H_g(e^{j\omega})$ is Gaussian shaped as shown in Figure 8 for the six values of σ used in our experiments.

Table 1. Table showing the orders M and lengths L of the Gaussian filters corresponding to $\sigma = 10, 20, 40, 60, 80, 100$.

σ	$M(=6\sigma)$	$L(=M+1)$
10	60	61
20	120	121
40	240	241
60	360	361
80	480	481
100	600	601

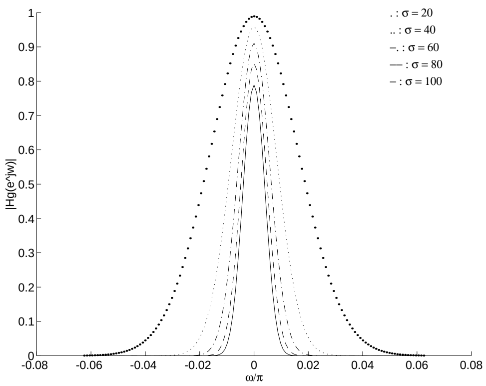

Figure 8. Plots of frequency responses $H_g(e^{j\omega})$ of Gaussian filters for $\sigma = 20, 40, 60, 80, 100$.

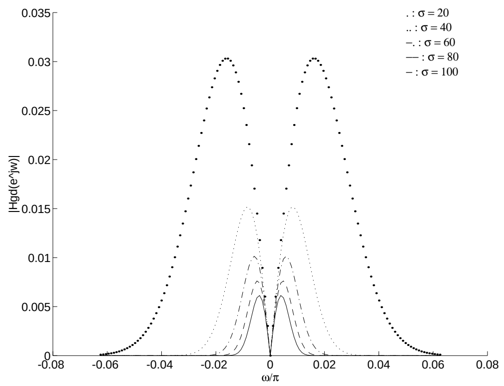

Figure 9. Equivalent frequency responses of a Gaussian filter in cascade with a bandlimited differentiator for $\sigma = 20, 40, 60, 80, 100$.

5.2.5. Differentiator

The differentiator is implemented as an FIR filter based on the frequency response of a bandlimited differentiator [24]:

$$H(e^{j\omega}) = \; j\omega \quad -\pi < \omega < \pi$$

The corresponding impulse response is:

$$h[n] = \begin{cases} 0 & n = 0 \\ \frac{cos \pi n}{n} & n \neq 0 \end{cases}$$

This infinite impulse response is multiplied by the Hanning window to obtain a finite impulse response $h_d[n]$.

It is interesting to note that the lowpass filtering operation of the Gaussian filter and the derivative operation of the differentiator may be combined to obtain a single filter with the derivative of the Gaussian function as its impulse response $h_{gd}(t)$:

$$h_{gd}(t) = g'(t) = -\frac{t}{\sqrt{2\pi}\sigma^3} e^{-\frac{t^2}{2\sigma^2}}$$

The corresponding frequency response is as follows:

$$H_{gd}(\Omega) = j\Omega e^{-\frac{\Omega^2\sigma^2}{2}}$$

This frequency response is plotted in Figure 9.

From the frequency response, we note that the combined operation is equivalent to bandpass filtering where σ controls the bandwidth of the bandpass filter.

Bandlimiting the signals tends to reduce noise, thus reducing the noise sensitivity problem associated with detecting zero crossing points. With increasing values of σ, the bandwidth of the bandpass filter decreases and therefore more noise rejection is achieved. This agrees with our expectation since larger values of σ correspond to coarser time scales.

5.3. Segmenter

The segmenter consists of two parts: a *function segmenter* for each function of the system, and a *qualitative behavior generator* to coordinate the whole segmentation process. The overall scheme is illustrated in Figure 10.

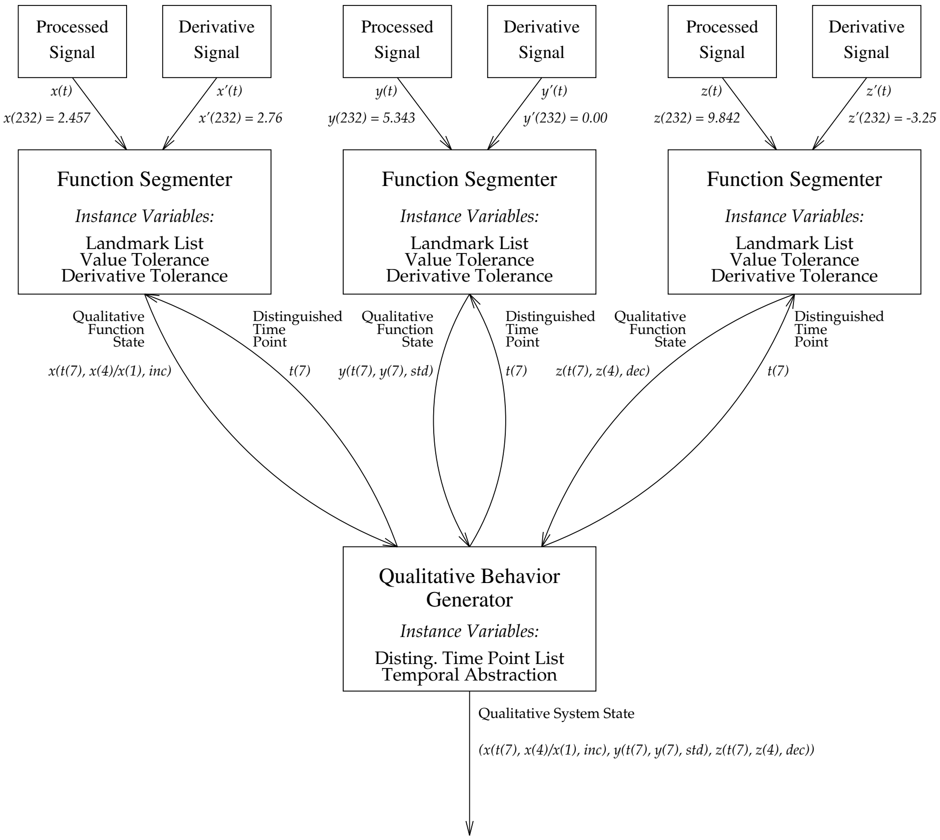

Figure 10. Overall scheme of segmentation to produce a qualitative behavior from processed signals and derivatives.

The function segmenter segments each signal at zero crossings of its derivative obtained from the differentiator. It then looks up its local landmark list to see if there is any existing landmark within a tolerance from the current value of the function. If so, the existing landmark becomes the qualitative value of the function in this state. If not, the segmenter creates a new landmark corresponding to the current value of the function, returns this landmark as the qualitative value of the function in this state, and stores the new landmark in the local landmark list. The direction of change of the function in the current state is obtained by observing the sign of the derivative. A positive derivative corresponds to *inc* (increasing). A negative derivative corresponds to *dec* (decreasing). A derivative within a tolerance from zero corresponds to *std* (steady). The qualitative value and the direction of change together form a qualitative state of the function.

The qualitative behavior generator keeps track of distinguished time points and coordinates the entire segmentation process. When any one or more of the function segmenters detects a zero crossing in their derivatives, the generator waits for τ sampling periods to see if any other segmenters also detect a zero crossing in their derivatives. The parameter τ therefore determines the level of temporal abstraction, as discussed in Section 5.2.3. The generator labels all times within these τ sampling periods as the same distinguished time point. It then signals *all* segmenters to segment their signals at this time point. The generator then collects a qualitative state of each function from its segmenter, and combines the qualitative states of all the functions of the system into a qualitative state of the system at the current distinguished time point. A series of such qualitative states form a *qualitative behavior* of the system. This is written into a text file for subsequent input into GENMODEL.

6. Results and Interpretation

The learning system was applied to data segments obtained from six patients during cardiac bypass surgery. [8] A data segment from each of the first five patients was used to study how qualitative models learned vary across patients. Six data segments from the same patient (Patient 6) during the same surgery were used to study how qualitative models learned vary within a patient. The models learned are compared with the cardiovascular model described in Section 4.4.

Each data segment was 1000 seconds (16.7 minutes) long, sampled at 1 Hz. The fault tolerance level in GENMODEL was set at 20% of the total number of qualitative states in each data segment. The operation performed in each case was to insert coronary artery bypass grafts, except in the case of Patient 2 which was to replace the aortic valve. Models were learned from the data segments at six different levels of temporal abstraction, represented by the six different values of L as shown in Table 1 in Section 5.2.4. The results for the data segment from Patient 5 and two of the six data segments from Patient 6 are described below. [14] includes results for the remaining data segments.

For each data segment, a brief overview of the patient's condition is given, followed by a plot of the original signals. Then the filtered signals at each of the six levels of

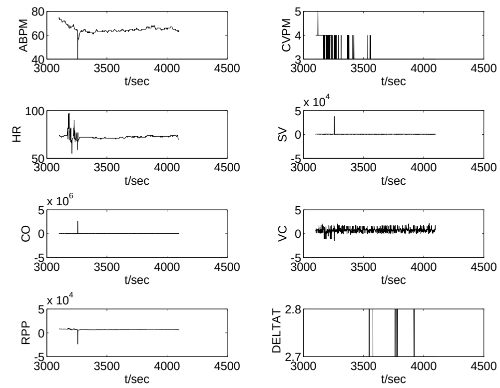

Figure 11. Patient 5: Original Signals. Note the relatively constant heart rate due to the effect of beta-blockers (see Section 6.3.4).

temporal abstraction is shown followed by the model learned and an interpretation of each of the model constraints.

In the following results, spurious constraints are not considered to be generally valid models of physiology but are supported by the example data, i.e. they are over-specific and likely to be lost as more examples come in (see Section 6.3.4).

6.1. Patient 5

The patient was a 66-year-old gentleman with a fairly long history of angina and a proven inferior infarct 3 months before the operation. He had very poor exercise tolerance, developing severe ischemia after very moderate exercise. His catheterization showed severe triple vessel disease with reasonably good left ventricular function. He was hypertensive and was treated with beta-blockers (Atenolol). He was also a non-insulin dependent diabetic.

The data segment was taken quite some time after the surgery had started. Before the period, the initial lightness of anesthesia caused a sharp rise in ABP from 90 mmHg systolic up to 160 mmHg systolic and that was sustained for several minutes. During the period, the depth of anesthesia (Enflurane) and analgesia (Alfentanil) and the dosage of GTN (glyceryl trinitrate or nitroglycerin, a vasodilator) were increased to bring the ABP back down.

L=61

$M^{-}(CVPM, \Delta T)$ (Spurious)
$M^{+}(SV, CO)$ Correct given that HR was constant due to beta-blockers.
$M^{+}(CVPM, \Delta T)$ (Spurious)

L=121

No constraints were obtained.

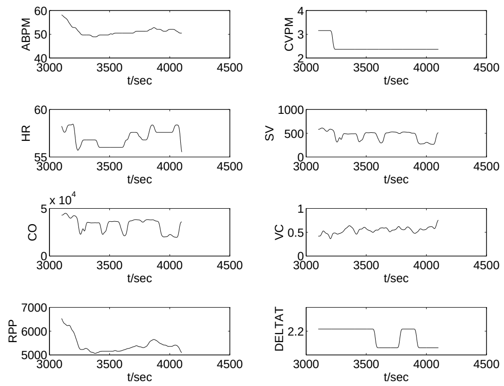

Figure 12. Patient 5: Filtered Signals ($L = 61$). Note that the trends of the relatively constant heart rate are amplified.

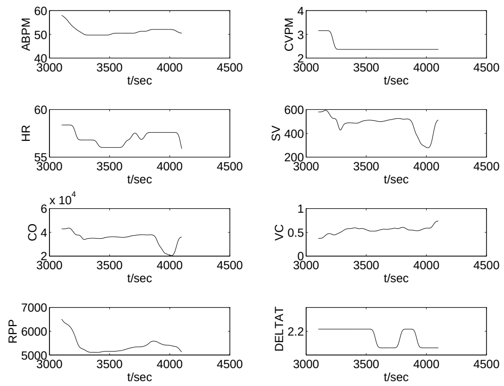

Figure 13. Patient 5: Filtered Signals ($L = 121$)

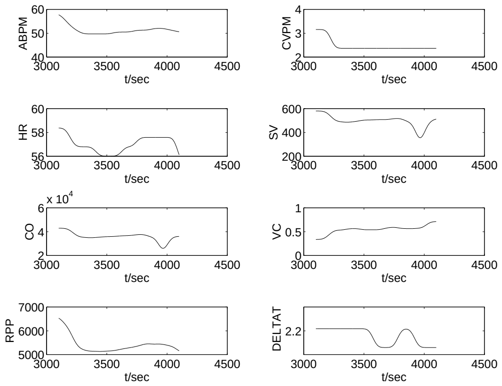

Figure 14. Patient 5: Filtered Signals ($L = 241$)

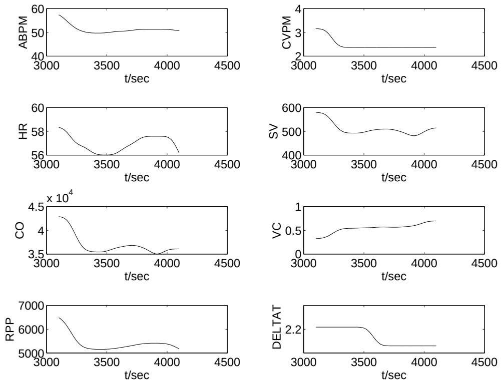

Figure 15. Patient 5: Filtered Signals ($L = 361$)

L=241

$M^+(SV, CO)$ Correct given that HR was constant due to beta-blockers.
$mult(HR, SV, CO)$ (Correct)

L=361

$M^+(ABPM, HR)$ (Spurious)
$M^+(ABPM, RPP)$ Correct given that HR was constant due to beta-blockers. ABPM dropped because of increased depth of anesthesia.
$M^+(HR, RPP)$ (Spurious)
$M^+(SV, CO)$ Correct given that HR was constant due to beta-blockers.
$mult(HR, ABPM, RPP)$ (Correct)
$mult(HR, CVPM, RPP)$ (Spurious)

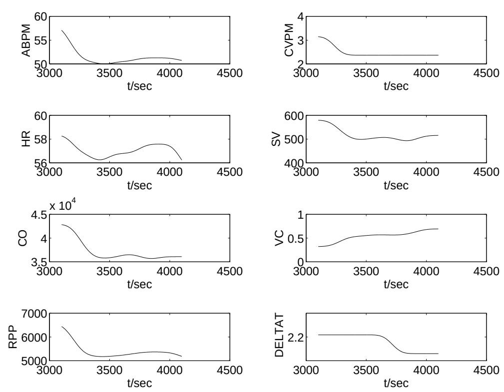

Figure 16. Patient 5: Filtered Signals ($L = 481$)

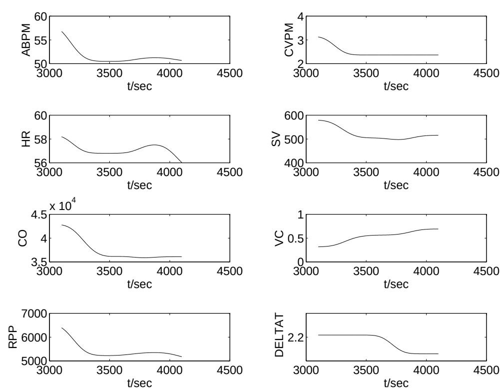

Figure 17. Patient 5: Filtered Signals ($L = 601$)

$mult(HR, SV, CO)$ (Correct)

L=481

$M^+(ABPM, RPP)$ Correct given that HR was constant due to beta-blockers. ABPM dropped because of increased depth of anesthesia.

$M^-(ABPM, VC)$ VC increased to compensate for decreasing ABPM due to increased dosage of anesthetic, analgesia and GTN.

$M^-(RPP, VC)$ (Spurious)

$mult(HR, ABPM, RPP)$ (Correct)

L=601

$M^+(ABPM, HR)$ (Spurious)

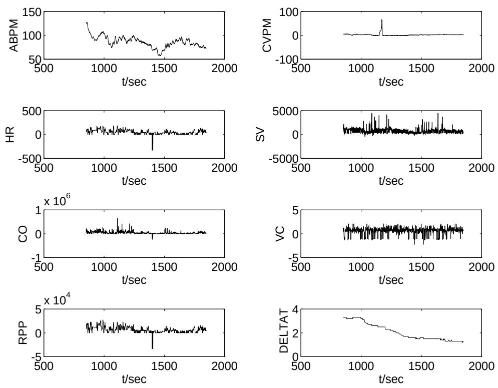

Figure 18. Patient 6, Segment 1: Original Signals

$M^+(ABPM, RPP)$ Correct given that HR was constant due to beta-blockers. ABPM dropped because of increased depth of anesthesia.
$M^+(CVPM, CO)$ Frank-Starling Law of the Heart.
$M^+(HR, RPP)$ (Spurious)
$mult(HR, ABPM, RPP)$ (Correct)
$mult(HR, CVPM, RPP)$ (Spurious)
$mult(HR, SV, CO)$ (Correct)

6.2. Patient 6

The patient was a 63-year-old gentleman having 1 internal mammary artery and 3 coronary artery grafts. He had a history of 6 years of hypertension, 4 years of angina and 20 years of chronic bronchitis. His angiogram showed well presented left ventricle, totally occluded right ventricle and severe disease at the origin of all left sided vessels. He was not on beta-blockers.

6.2.1. Segment 1

Previous to this segment, lightness in anesthesia caused rises in ABP (up to 180 mmHg systolic) at leg surgery, chest incision and sternotomy. The patient then developed myocardial ischemia. In response to this, the GTN dosage was increased, which along with hypovolemia caused the ABP to drop, with the result that ischemia improved at the expense of blood pressure. The depth of anesthesia was also increased.
L=61

$mult(HR, ABPM, RPP)$ (Correct)

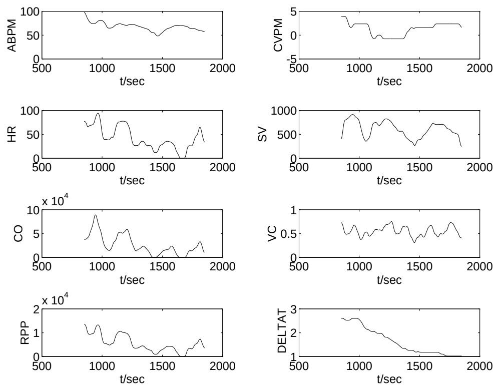

Figure 19. Patient 6, Segment 1: Filtered Signals ($L = 61$)

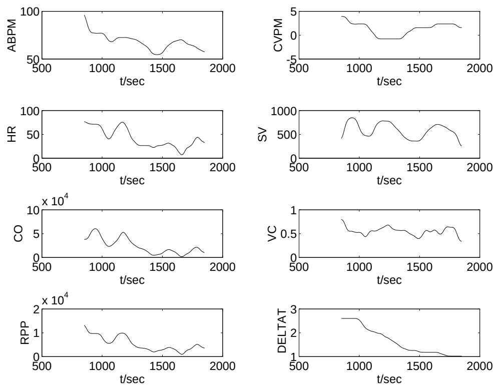

Figure 20. Patient 6, Segment 1: Filtered Signals ($L = 121$)

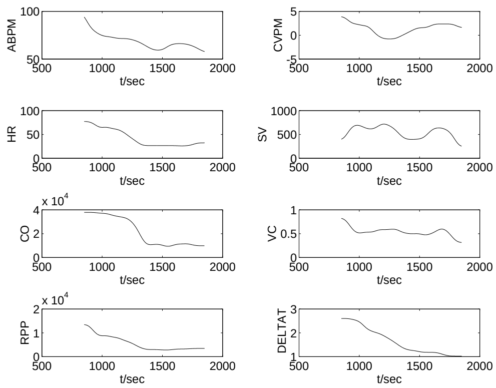

Figure 21. Patient 6, Segment 1: Filtered Signals ($L = 241$)

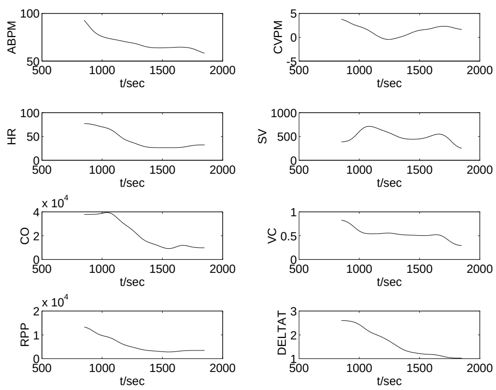

Figure 22. Patient 6, Segment 1: Filtered Signals ($L = 361$)

L=121

$mult(HR, ABPM, RPP)$ (Correct)
$mult(HR, SV, CO)$ (Correct)

L=241

$M^-(HR, SV)$ (Spurious)
$inv\_deriv(ABPM, RPP)$ (Spurious)
$inv\_deriv(ABPM, VC)$ (Spurious)

L=361

$M^+(HR, RPP)$ Both dropped because of increased depth of anesthesia.
$mult(HR, ABPM, RPP)$ (Correct)
$mult(HR, SV, CO)$ (Correct)

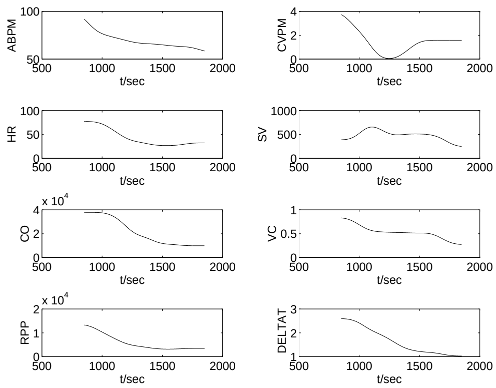

Figure 23. Patient 6, Segment 1: Filtered Signals ($L = 481$)

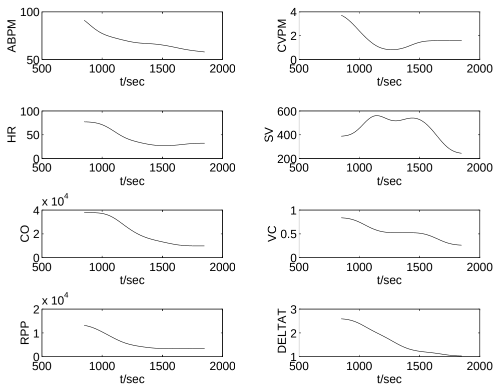

Figure 24. Patient 6, Segment 1: Filtered Signals ($L = 601$)

L=481

$M^+(HR, RPP)$ Both dropped because of increased depth of anesthesia.
$M^+(ABPM, \Delta T)$ Both dropped because of vasodilating effect of GTN.
$inv\_deriv(ABPM, RPP)$ (Spurious)
$inv\_deriv(ABPM, VC)$ (Spurious)
$mult(HR, ABPM, RPP)$ (Correct)
$mult(HR, ABPM, VC)$ (Spurious)
$mult(HR, CVPM, RPP)$ (Spurious)
$mult(HR, SV, CO)$ (Correct)

L=601

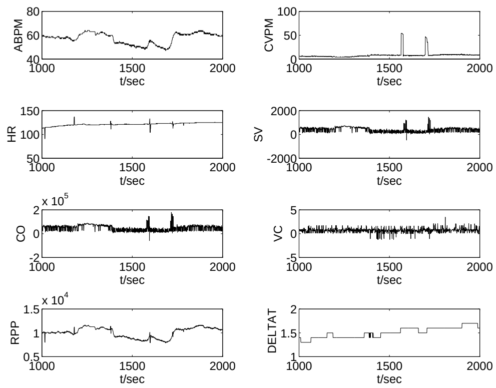

Figure 25. Patient 6, Segment 5: Original Signals

$M^+(ABPM, CO)$ Both dropped because of vasodilation and increased venous tone caused by increased GTN dosage.
$M^+(HR, RPP)$ Both dropped because of increased depth of anesthesia.
$M^+(ABPM, \Delta T)$ Both dropped because of vasodilating effect of GTN.
$M^+(CO, \Delta T)$ Both dropped because of vasodilation and increased venous tone caused by increased GTN dosage.
$inv\_deriv(ABPM, RPP)$ (Spurious)
$inv\_deriv(ABPM, VC)$ (Spurious)
$mult(HR, ABPM, RPP)$ (Correct)
$mult(HR, CVPM, RPP)$ (Spurious)
$mult(HR, SV, CO)$ (Correct)

6.2.2. Segment 5

The patient experienced low ABP post bypass due to poor cardiac performance secondary to a technically poor graft and possibly hypovolemia. Inotropic therapy (Dobutamine) was given which potentially caused the patient to develop myocardial ischemia. Blood infusion was performed to bring ABP back up.

L=61

$mult(HR, ABPM, RPP)$ (Correct)
$mult(HR, SV, CO)$ (Correct)

L=121

$mult(HR, SV, CO)$ (Correct)

L=241

$M^+(ABPM, CO)$ Both dropped initially because of poor cardiac performance and hypovolemia, and started to rise following blood infusion.

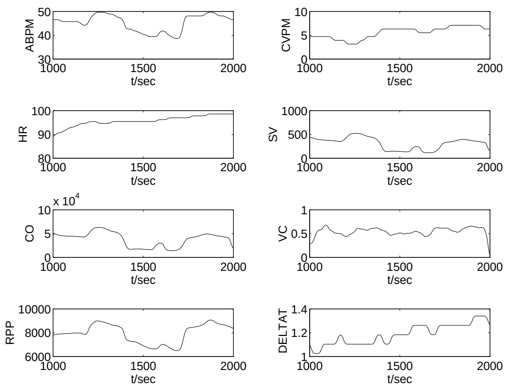

Figure 26. Patient 6, Segment 5: Filtered Signals ($L = 61$)

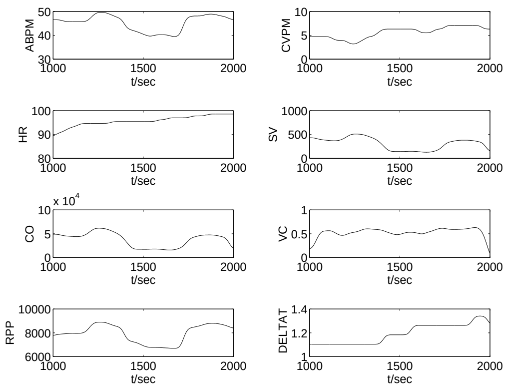

Figure 27. Patient 6, Segment 5: Filtered Signals ($L = 121$)

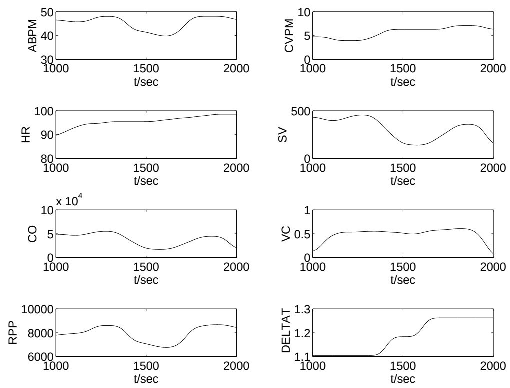

Figure 28. Patient 6, Segment 5: Filtered Signals ($L = 241$)

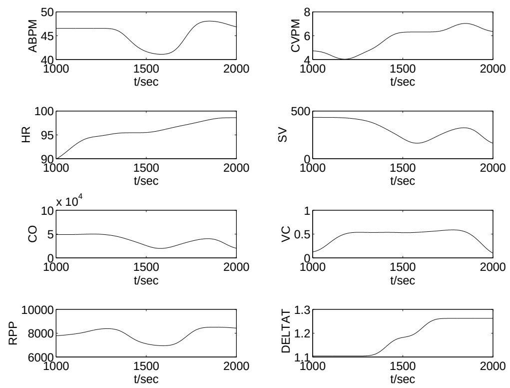

Figure 29. Patient 6, Segment 5: Filtered Signals ($L = 361$)

$M^{+}(ABPM, SV)$ Both dropped initially because of poor cardiac performance and hypovolemia, and started to rise following blood infusion.

$M^{+}(SV, CO)$

$mult(HR, ABPM, RPP)$ (Correct)

$mult(HR, SV, CO)$ (Correct)

L=361

$M^{+}(ABPM, CO)$ Both dropped initially because of poor cardiac performance and hypovolemia, and started to rise following blood infusion.

L=481

No constraints were obtained.

L=601

$M^{+}(SV, CO)$ Both dropped because of hypovolemia.

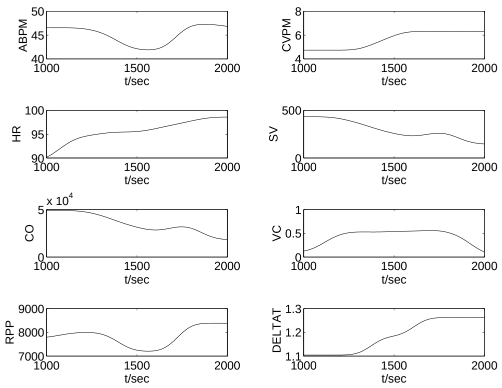

Figure 30. Patient 6, Segment 5: Filtered Signals ($L = 481$)

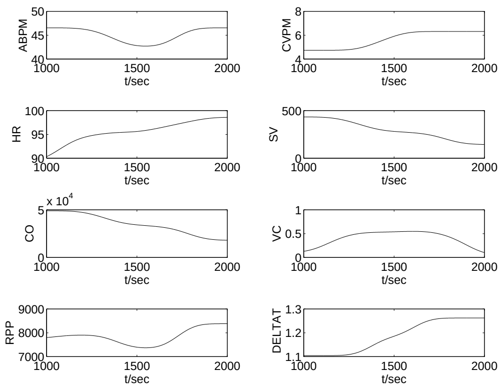

Figure 31. Patient 6, Segment 5: Filtered Signals ($L = 601$)

$M^{-}(HR, CO)$ HR increased both as a compensatory response to decreasing CO due to hypovolemia, and as a response to inotropic therapy.
$M^{-}(HR, SV)$
$inv\_deriv(SV, CO)$ (Spurious)
$mult(HR, ABPM, RPP)$ (Correct)
$mult(HR, SV, CO)$ (Correct)

6.3. Validity of Models Learned

The results in previous sections show that reasonable qualitative models can be learned from raw clinical data. Model constraints in our "gold standard" model constructed in Section 4.4 and other useful constraints showed up repeatedly in the models learned from our clinical data. These include:

- constraints valid in general such as $mult(HR, SV, CO)$ and $mult(HR, ABPM, RPP)$.
- constraints valid in specific patient conditions, possibly representing compensatory mechanisms, such as $M^{-}(HR, CO)$ and $M^{-}(CO, \Delta T)$ in hypovolemia.
- constraints valid under the effect of certain drugs. For example, $M^{+}(SV, CO)$ showed up in patients on beta-blockers because of their steady heart rate, and $M^{+}(ABPM, \Delta T)$ showed up in patients with an increased dosage of GTN causing vasodilation.

6.3.1. Model Variation Across Time

As discussed in Section 3, GENMODEL learns a qualitative model by creating an initial search space of all possible QSIM constraints, and successively pruning inconsistent constraints upon each given system state. Therefore if the system changes within the modeling period (in our case 16.7 minutes), resulting in a different underlying model, neither the old model nor the new model may be obtained. Constraints in the old model are pruned because they are inconsistent with the states after the system change. Constraints in the new model are pruned before the system change because they are inconsistent with the previous system. This may explain cases when we obtain very few or no model constraints. For example, if a patient is previously stable with an increasing relationship between the heart rate (HR) and the cardiac output (CO):

$$M^{+}(HR, CO)$$

but develops hypovolemia in the middle of a modeling process, resulting in a decreasing cardiac output and a compensatory mechanism involving an increasing heart rate, the new valid constraint is:

$M^-(HR, CO)$

But this will not appear in the final model because at the onset of hypovolemia, this constraint has already been pruned by GENMODEL according to states corresponding to the previously stable condition. Furthermore, the previously valid constraint $M^+(HR, CO)$ will be pruned because it is now inconsistent with the system states corresponding to hypovolemia. One may actually exploit this feature of model variation across time in the context of intelligent patient monitoring systems (see Section 7.1).

6.3.2. Model Variation Across Different Levels of Temporal Abstraction

The models learned in Sections 6.1 and 6.2 varied across different levels of temporal abstraction, represented by the filter length L. For example, constraints which involve the skin-to-core temperature gradient ΔT representing the level of vasoconstriction in the body, generally appeared only under large values of L, i.e. in coarser time scales. This means the response of ΔT generally lags behind the responses of other parameters. This may be due to the considerable heat capacity of the body causing a delay in the measurable effects of vasoconstriction.

In general, we observe that fewer model constraints were learned with decreasing L or finer time scales. This may be due to the following reasons:

- As discussed in Section 5.2.4, smaller values of L and therefore smaller values of σ correspond to larger cutoff frequencies in the lowpass Gaussian filters, and larger bandwidths in the bandpass filtering operation equivalent to the cascade of the Gaussian filter with the differentiator. This reduces the amount of noise rejection achieved, and results in noise sensitivity problems in detecting zero crossing points and therefore less accurate segmentation. This additional amount of noise may have caused correct constraints to be pruned, resulting in fewer or even no constraints left in the final model.

- Smaller values of L correspond to faster processes which may have more dynamic models. As discussed in Section 6.3.1, a system change within a modeling period can cause constraints belonging to both the previous and the current model to be pruned, resulting in a smaller model or even one with no constraints.

6.3.3. Model Variation Across Different Levels of Fault Tolerance

We observe that in general the size of the model learned increases with increasing levels of fault tolerance. A fault tolerance level of η means that GENMODEL allows for inconsistent states up to a fraction η of the total number of states in the system behavior before pruning a constraint. Therefore with larger η, fewer constraints will be pruned and the resulting model will contain more constraints.

An indication of η being set too high is that conflicting constraints start to appear. For example, in Patient 5 with $L = 61$ (Figure 12), both

$$M^{+}(CVPM, \Delta T)$$

and

$$M^{-}(CVPM, \Delta T)$$

appear in the model learned. This is because both $CVPM$ and ΔT are relatively steady and contain only few *inc* and *dec* segments which distinguish between the M^{+} and M^{-} constraints. Within a high level of tolerance, the distinction is obscured.

6.3.4. Sources of Error

False Positives: In the models learned, we observe that spurious constraints sometimes appeared in the resulting model. For example, in Segment 1 of Patient 6 ($L = 601$), we obtained the spurious constraint:

$$inv\_deriv(ABPM, RPP)$$

This may be due to several possible reasons:

- The waveforms are relatively smooth with few critical points. This results in a system behavior with few states, corresponding to few examples for learning. With the small sample size, it is relatively probable that these examples are consistent with the incorrect constraint. For instance, if whenever ABPM decreases, RPP is positive, then the above incorrect *inv_deriv* constraint will be learned. (In terms of the PAC learning model, the small sample size corresponds to a low accuracy and confidence level for learning. See Section 3.2.5.)
- The level of fault tolerance is set too high resulting in incorrect constraints not being pruned.

False Negatives: We observe that even constraints that are generally valid in all conditions, such as

$$mult(HR, SV, CO)$$

did not appear in every model learned. There are several possible reasons for this:

- Since few states are available in the data segment resulting in few examples for learning, if these examples are corrupted by noise, the correct constraint will be pruned.

- Values corrupted by noise are recorded as corresponding values by the system [14].
- The level of fault tolerance is set too low resulting in correct constraints being pruned.

Landmark Values: Temporal abstraction refers to how close two times have to be before we label them as the same distinguished time point. Similarly, we have to decide how close two function values have to be before we label them as the same landmark value. If the tolerance is set too low, we may amplify trends of relatively steady signals. This is the case in the heart rate signals of Patient 5 (Figure 12) which are relatively steady due to the effect of beta-blockers. The fluctuations within 2-3 beats per minute are amplified into a series of *inc* (increasing) and *dec* (decreasing) segments. The whole segment might well have been labeled as *std* (steady) if we had set the tolerance appropriately.

7. Conclusion and Further Work

The goal of this work is to develop a system for learning qualitative models from physiological signals. In Section 1 we mentioned two potential applications for such a system. First, the system could be a useful tool for knowledge acquisition from large amounts of data. Second, the system could be incorporated into an intelligent patient monitoring system to perform adaptive model construction for diagnosis in a dynamic environment. In the previous section, We have evaluated the performance of the system in knowledge acquisition and identified sources of error. Here, we examine its applicability in diagnostic patient monitoring.

From the models shown in Section 6, we see that constraints of models learned do track changes in patient condition over time. For example, the following changes were tracked:

Compensatory mechanisms during shock e.g. the constraints $M^-(HR, CO)$ and $M^-(CO, \Delta T)$ learned when the patient experienced hypovolemia.

Effects of drugs e.g. the constraint $M^+(SV, CO)$ tracked the effect of beta-blockers because of the patient's relatively constant heart rate, and the constraint $M^+(ABPM, \Delta T)$ tracked the effect of an increased dosage of GTN causing vasodilation.

7.1. A Learning-Based Approach to Diagnostic Patient Monitoring

Since model constraints learned track patient condition over time, we might be able to build a diagnostic patient monitoring system based on our learning system. The patient monitoring system continually learns models from patient data and detects changes in the models learned. Diagnoses are made based on these changes.

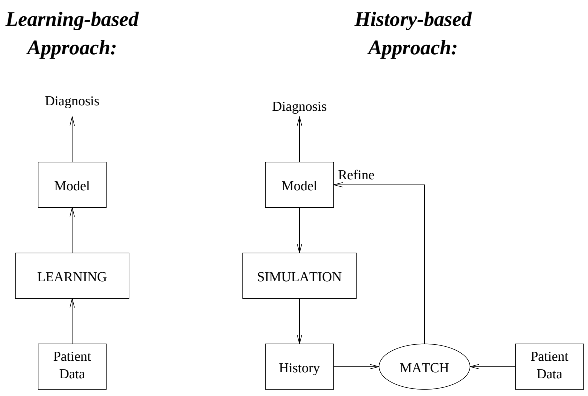

Figure 32. Two approaches to diagnostic patient monitoring. In the learning-based approach, models are continually learned from the patient data. In the history-based approach, a hypothesize-test-refine cycle is used to generate models that best match the patient data. In each approach, diagnoses are made based on the current model.

This learning-based approach to diagnostic patient monitoring is summarized in Figure 32.

The traditional history-based approach to diagnostic patient monitoring goes in the opposite direction. It generates histories based on different models. These histories are matched with the patient data. Diagnoses are based on models corresponding to histories that best match the patient data. Such a system would look for stability in constraints over time, recorded as some percentage of match to incoming data. It would attempt to detect when such measures changed, indicating that the constraint was no longer valid and that a new model was being generated by an altered patient state. This approach can be achieved by a hypothesize-test-refine cycle as shown in Figure 32 [6], [9].

The learning-based approach may be more efficient since the hypothesis model is generated directly from the patient data and there is no need for a hypothesize-test-refine cycle.

Acknowledgments

Part of the work described in this article was presented at the "AAAI Spring Symposium Series: Artificial Intelligence in Medicine" held at Stanford University in March 1994. This work is based on a thesis submitted by the first author in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering and Computer Science at the Massachusetts Institute of Technology in May 1994. This work was supported by and carried out at the Hewlett-Packard Laboratories in Bristol, U.K. The authors would like to thank Drs. Dave Reynolds and Roger Mark for helpful comments.

Notes

1. GENMODEL is implemented in UNSW Prolog V4.2 [5].
2. $A = \Omega(B)$ denotes B is a lower bound of A.
3. Monotone conjunctions are ones with positive literals only.
4. A clause is said to be *ground* if it does not contain any variables.
5. The front-end system and the segmenter are implemented in Objectworks\Smalltalk on the HP9000/720.
6. The Gaussian function has the smallest duration-bandwidth product with duration and bandwidth as defined in [31], and is therefore optimally localized in both the time and frequency domains.
7. In general, this property holds true for zero crossings obtained by applying *any* linear differential operator (including the Laplacian and the first derivative) to the filtered signal.
8. Raw data was recorded from the Hewlett-Packard Component Monitoring System at a local hospital. The eight parameters used for the experiments were derived from these primary measurements as described in Sections 4.2 and 4.3.

References

1. J. Babaud, A.P. Witkin, M. Baudin, and R.O. Duda. Uniqueness of the Gaussian kernel for scale-space filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):26–33, January 1986.
2. G. Balestra and D. Liberati. Qualitative simulation of urea extraction during dialysis. *IEEE Engineering in Medicine and Biology*, 11:80–84, June 1992.
3. A. Blumer, A. Ehrenfeucht, D. Haussler, and M.K. Warmuth. Occam's razor. *Information Processing Letters*, 24(6):377–380, April 1987.
4. I. Bratko, S. Muggleton, and A. Varšek. Learning qualitative models of dynamic systems. In *Proceedings of the International Workshop on Inductive Logic Programming*, pages 207–224, 1991.
5. E. Coiera. Generating qualitative models from example behaviours. DCS Report 8901, Department of Computer Science, University of New South Wales, Sydney, Australia, May 1989.
6. E. Coiera. Monitoring diseases with empirical and model generated histories. *Artificial Intelligence in Medicine*, 2:135–147, 1990. Revised version appeared in E. Keravnou (ed), Medical Artificial Intelligence I - Deep Models for Medical Knowledge Engineering, Elsevier, Amsterdam, (1992), 71-88.
7. E. Coiera. The qualitative representation of physical systems. *The Knowledge Engineering Review*, 7(1):55–77, 1992.
8. E. Coiera. Qualitative superposition of unmodelled systems. Technical Report HPL-92-166, Hewlett-Packard Laboratories, Bristol, UK, 1992.
9. E. Coiera. Editorial: Intelligent monitoring and control of dynamic physiological systems. *Artificial Intelligence in Medicine*, 5:1–8, 1993.
10. B.C. Falkenhainer and R.S. Michalski. Integrating quantitative and qualitative discovery: The ABACUS system. *Machine Learning*, 1:367–401, 1986.
11. F.L. Gobel, L.A. Nordstrom, R.R. Nelson, C.R. Jorgensen, and Y. Wang. Rate-pressure product as an index of myocardial oxygen consumption during exercise in patients with angina pectoris. *Circulation*, 57:549–556, 1978.
12. A.C. Guyton. *Textbook of Medical Physiology*. Saunders, Philadelphia, PA, sixth edition, 1981.
13. D.T. Hau. A waveform shape recognition system based on a blackboard architecture. Bachelor's thesis, Department of Electrical Engineering and Computer Science, MIT, June 1992.

14. D.T. Hau. Learning qualitative models from physiological signals. Master's thesis, Department of Electrical Engineering and Computer Science, MIT, May 1994.
15. L. Ironi, M. Stefanelli, and G. Lazola. Qualitative models in medical diagnosis. In E. Keravnou, editor, *Deep Models for Medical Knowledge Engineering*. Elsevier, Amsterdam, The Netherlands, 1992.
16. M.J. Kearns and U.V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA, 1994.
17. G. Kendon. An investigation of back-propagation for arterial blood pressure wave classification. Master's thesis, Department of Artificial Intelligence, University of Edinburgh, September 1992.
18. B. Kuipers. Qualitative simulation in medical physiology: A progress report. Technical Report MIT/LCS/TM-280, Laboratory for Computer Science, MIT, Cambridge, MA, June 1985.
19. B. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.
20. B. Kuipers. Qualitative simulation as causal explanation. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(3):432–444, May/June 1987.
21. D. Marr and E. Hildreth. Theory of edge detection. *Proc. Roy. Soc. London*, 207:187–217, 1980.
22. S. Muggleton, editor. *Inductive Logic Programming*. Academic Press, London, 1992.
23. T.E. Oh, editor. *Intensive Care Manual*. Butterworth, Sydney, Australia, third edition, 1990.
24. A.V. Oppenheim and R.W. Schafer. *Discrete-time Signal Processing*. Prentice Hall, Englewood Cliffs, NJ, 1989.
25. G.D. Plotkin. *Automatic Methods of Inductive Inference*. PhD thesis, University of Edinburgh, August 1971.
26. J.R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.
27. B.L. Richards, I. Kraan, and B.J. Kuipers. Automatic abduction of qualitative models. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 723–728, 1992.
28. R.L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.
29. L.J. Saidman and N.T. Smith, editors. *Monitoring in Anesthesia*. Butterworth, Boston, MA, second edition, 1984.
30. R.C. Schlant and R.W. Alexander, editors. *Hurst's the heart: arteries and veins*. McGraw-Hill, New York, 8th edition, 1994.
31. W.M. Siebert. *Circuits, Signals, and Systems*. The MIT Press, Cambridge, MA, 1986.
32. G. Strang. Wavelets and dilation equations: a brief introduction. *SIAM Review*, 31(4):614–627, December 1989.
33. P. Toal and J. Hunter. A qualitative model of cardiac haemodynamics. Technical Report AUCS/TR9001, Department of Computing Science, University of Aberdeen, Aberdeen, U.K., 1990.
34. L.G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.
35. A. Varšek. Qualitative model evolution. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, pages 1311–1316, Sydney, Australia, August 1991.
36. J. Weinberg, G. Biswas, and S. Uckun. Continuing adventures in qualitative modelling: A qualitative heart model. In *Proceedings of the Third International Conference on Industrial and Engineering Applications of AI Expert Systems (IEA/AIE 90)*, Charlestown, SC, 1990. ACM Press.
37. K.H. Wesseling, B. deWit, J.A.P. Weber, and N.T. Smith. A simple device for the continuous measurement of cardiac output. *Advances in Cardiovascular Physiology*, 5:16–52, 1983.
38. P.L. Wilkinson, J.V. Tyberg, J.R. Moyers, and A.E. White. Correlates of myocardial oxygen consumption when afterload changes during halothane anesthesia in dogs. *Anesthesia and Analgesia*, 59:233–239, 1980.
39. J.B. Wyngaarden, L.H. Smith, and J.C. Bennett, editors. *Cecil Textbook of Medicine*. Saunders, Philadelphia, PA, 19th edition, 1992.

40. A.L. Yuille and T.A. Poggio. Scaling theorems for zero crossings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):15–25, January 1986.