Monotone Pieces Analysis for Qualitative Modeling

Yuhong Yan [1] and **Daniel Lemire** [2] and **Martin Brooks**[3]

Abstract. It is a crucial task to build qualitative models of industrial applications for model-based diagnosis. A Model Abstraction procedure is designed to automatically transform a quantitative model into qualitative model. If the data is monotone, the behavior can be easily abstracted using the corners of the bounding rectangle. Hence, many existing model abstraction approaches rely on monotonicity. But it is not a trivial problem to robustly detect monotone pieces from scattered data obtained by numerical simulation or experiments. This paper introduces an approach based on scale-dependent monotonicity: the notion that monotonicity can be defined relative to a scale. Real-valued functions defined on a finite set of reals e.g. simulation results, can be partitioned into quasi-monotone segments. The end points for the monotone segments are used as the initial set of landmarks for qualitative model abstraction. The qualitative model abstraction works as an iteratively refining process starting from the initial landmarks. The monotonicity analysis presented here can be used in constructing many other kinds of qualitative models; it is robust and computationally efficient.

1 INTRODUCTION

Qualitative models are more suitable than classical quantitative models in solving many problems. Qualitative models are used in tasks such as diagnosis [18], explaining system behavior [9, 14], and designing novel devices from first principles [21]. Building a qualitative model for a complex system requires significant knowledge and is a time consuming process. Current research efforts are on automatic generation of qualitative models from numerical models . The numerical data originate either from simulation or sensors. Many qualitative models have been proposed with different tasks in mind, for example *Finite Relation Qualitative Model* (FRQ) [18], *Qualitative Derivative Model* (QD) [6], or *Qualitative Constraint Function* (QCF) [19]. Studying the different model abstraction processes, we find that the monotonicity of the numerical data plays an important role in constructing the qualitative models. Indeed, in [18, 22] where the FRQ is generated from simulation data by iteratively refining partitions of the data, the monotone pieces determine the initial landmarks for the refining procedure. Also, in [6], the analysis relies on "signs of deviation" [Δy] taking values in +, -, 0.Numerical Analysis provides some techniques, like linear regression and linear splines [13], but these methods are relatively expensive or approximate, so that better approaches are possible. [19] presents a machine learning approach which is even more expensive to use and has not shown to be able to deal with complex functions.

[1] NRC-IIT-Fredericton, Canada, email: yuhong.yan@nrc.gc.ca
[2] University of Quebec in Montreal, Canada
[3] NRC-IIT-Ottawa, Canada
In ECAI 2004 MONET Workshop on Model-Based Systems, Valencia, Spain. (NRC code 47159).

This paper presents one robust and inexpensive solution to the problem of piecewise monotone segmentation of numerical data. Scale-dependent monotonicity as defined in [2] "ignores" numerical differences smaller than a given scale or "tolerance" δ. For a given scale $\delta > 0$, the numerical data can be partitioned into intervals upon each of which the function is "quasi-monotone" or δ-monotone. Compared with many common numerical segmentation methods, δ-monotonicity analysis is computationally efficient and scalable ($O(n)$), and the number of monotone segments are invariant under different implementations, which makes it suitable for constructing qualitative models. The physical meaning of δ is the magnitude of noise tolerated by the qualitative model. Based on δ-monotonicity analysis, we develop an automatic method to abstract an FRQ model from simulation data.

2 MONOTONICITY ANALYSIS FOR CONSTRUCTING QUALITATIVE MODELS

Monotonicity analysis is critical for constructing some types of qualitative models. This section shows three examples, FRQ, QD and QCF, among which FRQ and QD are used for model-based diagnosis and QCF is used for explaining system behavior. Because FRQ will be used in section 4, we describe it in detail.

The Finite Relations Qualitative (FRQ) Model is a mapping between two sets of intervals:

Definition 1 *A **Finite Relation Qualitative** model q is a mapping between two sets of intervals* Ψ_1 *and* Ψ_2*:* $q : \Psi_1 \mapsto \Psi_2$.

The source set $\Psi_1 = [lm_i^1, lm_{i+1}^1]$ is the set of qualitative inputs to the model, the target set $\Psi_2 = [lm_k^2, lm_{k+1}^2]$ is the set of qualitative outputs from the model, the values lm_i are called *landmarks*.

The mapping is a multiple-valued relation, i.e. $\forall [lm_i^1, lm_{i+1}^1] \in \Psi_1$, $q : [lm_i^1, lm_{i+1}^1] \mapsto \Psi_2^{'} \subset \Psi_2$, where $\Psi_2^{'}$ is empty or has one or more elements. If we can enumerate the mapping pairs, q can be presented in set of tuples, as: $\{([lm_i^1, lm_{i+1}^1], [lm_j^2, lm_{j+1}^2]), \ldots, ([lm_p^1, lm_{p+1}^1][lm_q^2, lm_{q+1}^2])\}$

Definition 1 can be extended to the multi-dimensional case. The intervals used here are closed, unlike the traditional qualitative presentation as in [14]. Use of open versus closed intervals depends on the problem. . Other than in the Qualitative Simulation (QSIM) [14], where the landmarks are the solutions or other significant values of the Qualitative Differential Equations (QDE), the real valued landmarks in our application of model-based diagnosis do not have significant meanings. And the intervals are treated as strings in the diagnosis engine. Thus open or closed intervals make no difference.

From the numerical model $y = f(x)$ to the qualitative model, we need a qualitative model abstraction procedure:

Definition 2 *A system* $y = f(x)$ *with ordered landmarks* $x_1, x_2, \ldots, x_n \in \mathrm{Domain}(f)$, *and* $y_1, y_2, \ldots, y_m \in$

$\mathrm{Range}(f)$. *A* ***qualitative abstraction*** $q(f)$ *is a mapping from set* $\Psi_1 = [x_1, x_2], [x_2, x_3], \ldots, [x_{n-1}, x_n]$ *to set* $\Psi_2 = [y_1, y_2], [y_2, y_3], \ldots, [y_{n-1}, y_n]$*:*

$$q(f): \quad [x_1, x_2], [x_2, x_3], \ldots, [x_{n-1}, x_n] \mapsto \\ [y_1, y_2], [y_2, y_3], \ldots, [y_{n-1}, y_n],$$

such that $\forall [x_i, x_{i+1}]$, $q(f) : [x_i, x_{i+1}] \mapsto [y_j, y_{j+1}]$ *iff* $\exists x, y$ *such that* $f(x) = y$ *and* $x_i \leq x \leq x_{i+1}, y_j \leq y \leq y_{j+1}$*. A qualitative abstraction* $q(f)$ *provides a qualitative model.*

Going from f to $q(f)$ is a discretization procedure [18, 22]. If f is monotone, we can determine the landmarks as the corners of the bounding rectangle of f; that is, the smallest rectangle $\square = [x_a, x_b] \times [y_a, y_b]$ such that $y = f(x) \bigwedge x_a \leq x \leq x_b \Rightarrow (x, y) \in \square$. But robustly detecting monotone pieces from discrete simulation data is not a trivial problem. [18] limits the algorithm to monotone functions or monotone segments of a function. With the monotonicity constraint, the tuples are determined by the bounding landmarks (e.g. the corners of the rectangle). However, they [18] assume that monotonicity can be determined from the value of derivatives computed numerically from the discrete simulation data. We argue that a derivative-based approach is not sufficiently robust, because discrete simulation data may contain numerous small-amplitude numerical disturbances, which will result in many meaningless monotone segments. [22] uses random points to probe the extrema between the intervals; this pragmatic solution assumes a large number of random points. The extrema can be missed and the abstracted qualitative behavior may not be "sound".

[6] presents a method to abstract qualitative deviation models (QD) from numerical models. If f is monotone, the sign of deviation $[\Delta y]$ from a reference point x_{ref} is defined as +,-, or 0, according to whether f is increasing, decreasing or flat. For example, if f is monotone increasing, we have $[\Delta y] = \mathrm{sign}(f(x) - f(x_{ref})) = \mathrm{sign}(x - x_{ref}) = [\Delta x]$. If f is not monotone, [6] suggests that the domain has to be split in sub-intervals so that f is monotone on each sub-interval. However, monotonicity is a strong requirement, even on sub-intervals. It may result in a large number of monotone pieces, due to numerical disturbances caused by noise or computational round-off errors.

The two approaches discussed above illustrate the importance of monotonicity analysis and the associated problem of robustness; we now discuss a third approach having more robustness but lacking in scalability. Inductive learning is used in [19] to automatically construct qualitative models from quantitative examples. The monotonicity definition in [19] has the same spirit as partial derivative: the function is strictly increasing (decreasing) in its dependence on the i-th variable. The induced qualitative model is a binary tree, called a qualitative tree, which contains internal nodes (called splits) and qualitatively constrained functions at the leaves. A split is a partition of a variable. The qualitative constraint functions (QCF) define qualitative constraints on the function's range. The learning algorithm QUIN, a top-down greedy approach similar to ID3, is used. Given examples as training data, QUIN chooses the best split by comparing the generated partitions: for every possible split, it splits the examples into two subsets, finds the best QCF in both subsets, and selects the split which minimizes the tree error-cost (a utility function). It puts the best split at the root of the tree and recursively finds subtrees for the corresponding subsets, until the best split does not improve the tree error-cost. QUIN is an unsupervised learning algorithm. It determines the landmarks for the splits. QUIN claims to handle noisy data, and at least in simple domains, produces qualitative trees that correspond to human intuition. However, the training data in QUIN is composed of all possible pairs of points from the quantitative data. Thus, for n points, the learning set has $n(n-1)$ samples, thus constraining how big n can be. The example in [19] uses only 12 points to estimate three segments of linear functions, which gives 4 points to each linear segment. It is dubious that with such limitation on the sampling rate, this method is able to deal with more complex functions. Discrete data from a simulation environment may contain thousands of data points. The efficiency of QUIN on simulation data is not proven.

In response to the problems of robustness and scalability that plague other methods, the next section presents a method for robustly partitioning scattered data into monotone segments in linear time.

3 SCALE-DEPENDENT MONOTONICITY

3.1 $\delta-$monotonicity

Given a finite series of measurements $x_1, x_2, \ldots, x_n$ over an interval I (a "time series"), we want to partition I into a small number of subintervals where for each subinterval the measures either go up or down, in a general sense. Then, qualitatively, we can describe the measurements over each subinterval as a range of measured values which either increase or decrease. Of course, we desire noise tolerance, so that small perturbations do not result in numerous small subintervals. For example, we might say that over a given interval where values generally increase, it is acceptable for the measure to drop by δ once, where δ is some small parameter. Also, we want to be able to compute these segments on-line, in time $O(n)$, using very little memory ($O(1)$). For example, Fig. 1 depicts a data set made of a few "$\delta-$monotone" segments.

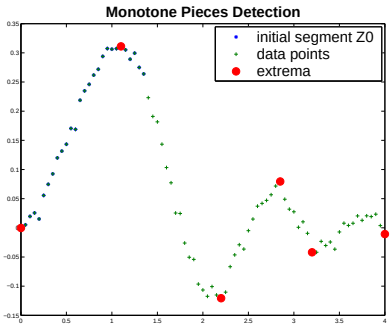

Figure 1. Plot of measurements where the end points of each $delta-$monotone segment are indicated by large filled circles. As can be seen on the figure, we tolerate small variation in the general upward or downward trend. The function is $e^{-x}\sin(x^2)$ with white noise having a standard deviation of 0.01. The parameter δ for this example is 0.05. Z0 and extrema are computed by our algorithms.

We want to be able to determine at any given point whether the measurements are going up or down in a robust fashion. One might be tempted to first use linear splines to approximate the data [13], and then segment the domain based on the sign of the slope in a neighborhood of the data point. As long as the spline function remains relatively close to the data points, this should offer a reasonable indication as to whether the measures are going up or down. Alternatively, we could use linear regression to find segments that are

closely approximated by a straight line; that is, we no longer require approximation by a continuous linear spline, but only piecewise approximation by straight lines forming a possibly discontinuous function. However, such linear fitting algorithms are relatively expensive or approximate [13]. Specifically, no known $O(n)$ algorithm finds a linear approximation with the least number of segments given some tolerance δ.Moreover, as can be seen in Fig. 1, the *function* resulting from the measurements can have some *curvature*, forcing the curve fitting algorithm to use several segments even if the measures are consistently going upward (or downward). Hence, the linear spline approach will lead to needlessly expensive algorithms.

A naïve approach to the segmentation problem might be as follows. Given an ordered set of measurements $\{x_k\}$ and some small tolerance value $\delta > 0$, we say that the data points are *not going down* or is *upward monotone*, if consecutive measures do not go down by more than δ, that is, are such that $x_i - x_{i+1} > \delta$. However, this definition is not very useful because measures can repeatedly go down and thus the end value can be substantially lower than the start value. A more useful definition of *upward monotonicity* would be to require that we cannot find two succesive measures x_i and x_j ($j > i$) such that x_j is lower than x_i by δ ($x_i - x_j > \delta$). This definition is more useful because in the worse case, the last measure will be only δ smaller than the first measure. However, we are still not guaranteed that the data does in fact increase at any point. Hence, we add the constraint that we can find at least two successive measures x_k and x_l ($l > k$) such that x_l is greater than x_k by at least δ ($x_l - x_k \geq \delta$).

To summarize, given some value $\delta > 0$, we say that a sequence of measures is *upward δ-monotone* if no two successive measures decrease by as much as δ, and at least one pair of successive measures increases by at least δ. Similarly, we say that a set of measures is *downward δ-monotone* if no two successive measures increase by as much as δ, and at least two measures decrease by at least δ.

Observe that the proposed definition for robust monotonicity doesn't depend on the x-axis metric (distance between measured events), but only on the metric on the measured values. Linear splines, on the other hand, would require metrics on both the x-axis and the measured values, which is another argument for using our simpler definition. Then again, with the proposed definition, it is possible to use the x-axis metric to eliminate segments that are too small, but doing so does not complicate the algorithms and can be done at a later stage.

This generalized definition of monotonicity was introduced in [2] using $\delta-$pairs: a $\delta-$pair is a pair of successive data points x_i, x_j ($j > i$) such that $|x_i - x_j| \geq \delta$, and for any k such that $i < k < j$ then $|x_k - x_i| < \delta$ and $|x_k - x_j| < \delta$ (see Fig. 2). The direction of a $\delta-$pair is *upward* (positive) if $x_j > x_i$ and *downward* (negative) if $x_i > x_j$. Notice that $\delta-$pairs having opposite directions cannot overlap.

A sequence of successive measurements x_i, x_{i+1}, ..., x_j is $\delta-$monotone if it contains at least one $\delta-$pair and all $\delta-$pairs have the same sign. We say that the measurements are *upward* $\delta-$monotone (positive) if all $\delta-$pairs are positive and *downward* $\delta-$monotone (negative) if all $\delta-$pairs are negative. A $\delta-$structure is a disjoint partition of the measurements into $\delta-$monotone sets of alternating sign. For a given sequence of measurements, there may be several $\delta-$structures: think of a curve going up, then flat, and then down again – does the flat part of the curve belong to the first or last segment? However, despite some degree of freedom in setting the boundaries between $\delta-$monotone segments, all $\delta-$structures have the same number of segments. A $\delta-$structure can be computed in linear time ($O(n)$) and constant space ($O(1)$) by scanning sequentially through the measurements, starting a new segment every time we encounter a $\delta-$pair having sign opposite to that of the current segment.

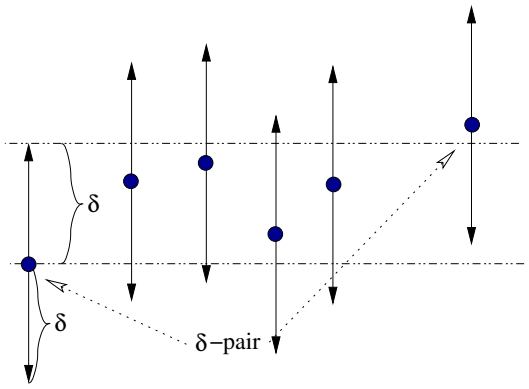

Figure 2. Example of a $\delta-$pair: data points are circles; the arrows indicate the size of δ.

3.2 Multidimensional Case

In a multidimensional case, we can fix all of the dimensions but two and define monotonicity over the remaining two degrees of freedom using the above approach; a similar approach is used to define monotone curves [1, 7, 10]. However, it may be preferable to look at more general definitions of monotonicity over multidimensional spaces [3–5, 11, 15, 20].

3.3 Choosing the Tolerance δ

As the next proposition shows, $\delta-$monotonicity is robust under white noise. As long as δ is chosen to be sufficiently larger than the noise level, it is unlikely that we will detect spurious segments.

Proposition 1 *Given a pair measured values $x_1 < x_2$ with $x_2 - x_2 \geq \delta$, then for any tow independently drawn samples $epsilon_1, epsilon_2$ from the normal distribution $N(0, \sigma)$, the probability that $x_1 + \epsilon_1 > x_2 + \epsilon_2$ is $P(|N(0,1)| > \frac{\delta}{\sqrt{2}\sigma})$.*

Proof. The proof relies on the fact that the subtraction or addition of two normal distributions with variance σ^2 is a normal distribution with variance $2\sigma^2$ [16]. Since $N(0,\sigma) - N(0,\sigma) = N(0,\sqrt{2}\sigma)$, we have

$$P(|N(0,\sigma) - N(0,\sigma)| > \delta) = P(|N(0,\sqrt{2}\sigma)| > \delta)$$

and $P(|N(0,\sqrt{2}\sigma)| > \delta) = P(|N(0,1)| > \frac{\delta}{\sqrt{2}\sigma})$. □

Notice that $P(|N(0,1)| > \frac{\delta}{\sqrt{2}\sigma})$ goes down to zero exponentially as the noise level σ goes down, hence the robustness. For a more general results, using n samples, we would need to ask for the probability that $x_n + \epsilon_n > \max\{x_1 + \epsilon_1, \ldots, x_{n-1} + \epsilon_{n-1}\}$ or simply $\epsilon_n > \max\{\epsilon_1, \ldots, \epsilon_{n-1}\}$. However, $\max\{\epsilon_1, \ldots, \epsilon_{n-1}\}$ is given by the Gumbel Extreme Value Distribution. Detailed analysis is left for a future paper.

One problem that arises when constructing a qualitative model is the choice of δ . If δ is chosen too small, then the resulting $\delta-$structure has many segments, not much noise resilience, and the qualitative model becomes too complex; if δ is chosen too large, then the $\delta-$structure does not capture the piecewise monotone behavior of

the data upon which it is based, and the model is an oversimplification.

When solving this problem, one may gain insight by considering the range of $\delta-$structures, as δ varies from 0 to ∞. At scale $\delta = 0$, the $\delta-$structure's $\delta-$monotone segments are exactly the data's (ordinary) monotone segments. For measurement sequence F, let $|F| = \max F - \min F$. For $\delta > |F|$, the $\delta-$structure has no alternating segments, i.e. all information about F has been lost. As δ moves upward within the interval $0 \leq \delta \leq |F|$, the number of alternating segments in F's $\delta-$structure decreases monotonically. For a given δ, when the $\delta-$structure has N alternating segments, then there exists $\epsilon > 0$ such that the $(\delta + \epsilon)-$structure has N alternating segments, but this is not necessarily the case for F's $(\delta-\epsilon)-$structure. In other words, the set of all δ for which F's $\delta-$structures have the same number of alternating segments is a half-open interval, closed below and open above. Thus, there is a least δ for which F's $\delta-$structure has no more than any given number of alternating segments. This δ is a *critical delta* for F. F has finitely many critical δs; they constitute the entire set from which one needs to pick when choosing δ for construction of a qualitative model. The set of all critical deltas for F can be computed in constant space with time proportional to the square of the number of data points in F.

In other words, if we don't know which δ to choose, presumably because we have no good estimate of our noise level, but we have some idea of the expected number of segments, then we can adjust δ so that we have at most the number of expected segments. This method will work well if the noise does not have any significant spike of an amplitude such that it creates false segments even for relatively large δs; hence it will work well if the noise is mostly white noise.

3.4 Joint $\delta-$Structures

Suppose we have several sequences of measurements $F, G, H, \ldots$ over the same interval. For a given δ, we have $\delta-$structures for each one of those. Given several partitions of an interval, a joint a partition is a partition such that its closure under unions contains all sets of all partitions. We can find a joint partition simply by considering the closure of the sets under intersection. However, because $\delta-$structures are not unique, it is possible for different joint partitions to contain different numbers of segments depending on the choice of $\delta-$structures. However, when considering all possible joint partitions, there are joint partitions among them having a minimal number of segments. We say that a joint partition of the $\delta-$structures is a joint $\delta-$structure if it contains this minimal number of segments. In this sense, the number of segments in a joint $\delta-$structure is independent of the algorithm or specific implementation used. A linear time algorithm to compute a joint $\delta-$structure is obtained by simultaneously computing $\delta-$structures for each of measurement sequences such that segments end at the same sample whenever possible.

Based on the above analysis, we have developed two algorithms. The first returns the initial $\delta-$monotone segment, the second one computes the remaining $\delta-$monotone segments. Due to space constraints, they are omitted. Figure 1 illustrates the results of the two algorithms on one function.

4 BUILDING FINITE RELATION QUALITATIVE MODEL FOR DIAGNOSIS USING δ-MONOTONE ANALYSIS

A qualitative model is a description of the system that covers all physically possible situations and that is as concise as possible with respect to the purpose of model-based problem solving. In this section, we develop a method to abstract FDQ models from simulation data for model-based diagnosis. Model-based diagnosis uses a model of normal system operation, and infers the faulty components which cause a discrepancy between the model's predicted measurements and those observed by sensors. The diagnosability of a fault is determined by the different projections of the faulty behavior and the normal behavior on the observable variables [8]. On one hand, a qualitative model should be fine enough to distinguish the two behaviors. On the other hand, the qualitative model should be abstracted enough to include only the information relevant the diagnosis task context [17, p.56]. As we know, a generic qualitative model for diagnosing all faults in all contexts does not exist [18], since this would require infinite information. Therefore the solution is to design a method to abstract a qualitative model for diagnosing a given set of faults in given system contexts.

4.1 Model Abstraction based on Detectability Analysis

Detectability analysis answers the question whether and under which conditions a faulty and normal system behavior can be discriminated. Intuitively, a fault is discriminable when the faulty behavior is disjoint from the normal behavior, and the discrepancy is projected on some observables. [8] categories the relations of the two modes into three classes: non-discriminable (ND), deterministically discriminable (DD) and possibly discriminable (PD). A fault is deterministically discriminable if the projection of the faulty mode on some observables is disjoint with the projection of the normal mode on the same set of observables:

$$SIT_{DD} : Proj_{\{obs_1,\ldots,obs_i\}}(R_r) \cap Proj_{\{obs_1,\ldots,obs_i\}}(R_f) = \phi \quad (1)$$

In (1), R_r and R_w are the normal and faulty models respectively. $\{obs_1, \ldots, obs_i\}$is the set of observables sufficient to detect the fault, a subset of the total collection of observables. Under certain conditions, if there is at least one observable such that the projections of the two modes can be differentiated, the fault is detectable.

The model abstraction is a iteratively refining process. It starts from no partition, i.e. the variables take values from the whole domains. The abstracted model is the coarsest in this case. If detectability is not satisfied, the domain of a character variable (as in [8]) is split into two sub-intervals by a newly generated landmark. And the other variables are partitioned accordingly. In this manner, a finer relation is derived. The detectability analysis is executed for each subdivision. This procedure continues recursively until detectability is satisfied within at least one subdivision, or the partition is finer than a preset threshold (cf. section. 4.5).

The character variables are those variables causing the greatest changes to the observables. Special techniques are needed to identify character variables. The causal ordering algorithm [12] can be used for this purpose. In this paper, the character variables are chosen manually. The selection of the minimal observables to detect the fault is the sensor placement problem. Though we can combine the sensor placement analysis with the abstraction procedure, we prefer to keep the problem simple and concentrate only on the modeling issues in this paper.

4.2 Qualitative Relation on δ-Monotone Segments

Section 3 presented the concept of δ-monotonicity. On a δ-monotone segment, the qualitative relation is determined in the following way.

Assume known relation $v_j = f(v_i)$, where v_i has two adjacent landmarks $lm_p^{v_i}$ and $lm_{p+1}^{v_i}$; the landmarks of v_j can be determined by the bounding rectangle:

$$lm_{q+1}^{v_j} := max(f(v_i)),\ lm_p^{v_i} < v_i < lm_{p+1}^{v_i} \quad (2)$$

$$lm_q^{v_j} := min(f(v_i)),\ lm_p^{v_i} < v_i < lm_{p+1}^{v_i} \quad (3)$$

For simulated measurements, the noise is so small that the simulation data can be considered close enough to the true behavior of the system; thus the above formulas are sound abstractions. However, when using actual sensor data, the true behaviors may be disguised by the noise. Depending on the nature of the noise, different treatments are adapted. If the noise is low-amplitude, we can use the joint δ-structure to give the best estimation of the behavior from several measurements(cf. section 3.4), i.e identify the largest scale δ such that each measurement sequence's δ-structure has the same number of alternating segments; we then use the average value and location of the corresponding segments' respective endpoints. Large-amplitude outliers need to be removed prior to δ-monotone analysis.

For the multiple dimensional case, the partition of v_i is executed safely at segments such that v_i-v_j is δ-monotone, where $i \neq j$. This condition gives the initial landmarks (cf. section 4.3).

4.3 Initial Landmarks

For a system with behavior $R(\mathbf{V})$, the partition of v_i is executed safely using segments such that every pair v_i-v_j is δ-monotone, where $i \neq j$. If v_1 is chosen as the character variable to be split, the initial landmarks of v_1 are determined by the segments of all v_1-v_j pairs. For each pair of variables v_1-v_j, δ-monotonicity gives landmarks for v_1, $\{lm_{p,\{v_1,v_j\}}^{v_1}\}$, the overall landmarks for v_1 is the union of the sets:

$$S := \cup\{lm_{p,\{v_1,v_j\}}^{v_1}\} \quad (4)$$

4.4 Parameter Changes

In order to cover all the behavior, we need to simulate situations when some parameters change; for example, changes of external temperature. We get a group of curves in this case. These curves compose a shape. For an FRQ model, the qualitative description of the shape is similar to the description of a curve, i.e. we express the shape as the bounding rectangle. Then only the two envelope curves (say, c1 and c2) determine the corners of the bounding rectangle. Assume c1 and c2 are δ-monotone at $[lm_i, lm_{i+1}]$ for variable v_i, the landmarks for v_j are calculated as:

$$\begin{aligned} lm_{q+1}^{v_j} &:= max(lm_{q+1}^{v_j,c1}, lm_{q+1}^{v_j,c2}) \\ lm_q^{v_j} &:= min(lm_q^{v_j,c1}, lm_q^{v_j,c2}) \end{aligned}$$

where $lm_{q+1}^{v_j,c1}$, $lm_{q+1}^{v_j,c2}$, $lm_q^{v_j,c1}$, $lm_q^{v_j,c2}$ are calculated as in section 4.2.

4.5 Algorithm of Qualitative Model Abstraction

Integrating our solutions to all the issues described above, we designed the following algorithm to abstract the qualitative model (see Algorithm 1). In the algorithm, p is a tree structure to record the partitions of character variables. The nodes on one layer record the partitions of one character variable. For example, assume v_1 is the first character variable, each node on the first layer has an interval $[lm_p^{v_1}, lm_{p+1}^{v_1}]$. The partitions of the children are the sub-partitions of their parent. For example, assume the parent $[lm_p^{v_1}, lm_{p+1}^{v_1}]$ has a node $[lm_q^{v_2}, lm_{q+1}^{v_2}]$, at the child, the model is partitioned fist by $[lm_p^{v_1}, lm_{p+1}^{v_1}]$, then by $[lm_q^{v_2}, lm_{q+1}^{v_2}]$. The leaves of the tree are labeled by the detectability in the partitioned region: DD means deterministic diagnosable and ND means none diagnosable. If any leaves are labeled DD or the partition is finer than threshold, the algorithm stops refining the relation, and computes the qualitative relations at each partition. Adjacent ND partitions are combined to simplify the results. Notice that the refining procedure separates the DD region gradually; we do not see "possible diagnosable" situations during the refinement. (There is no "possible diagnosability" here.) The preprocessing of dynamic models is not included in the algorithm. For simplicity, the algorithm does not include treatment for parameter changes.

Algorithm 1 Qualitative Model Abstraction.

- V_{char}: the set of character variables
- V_{obs} : the set of observables
- R_r, R_f: the simulated normal and faulty behaviors.
- p: a tree to record partition of character variables
- D: the threshold of minimum interval

for one $v_i \in V_{char}$ **do**
  calculate initial landmarks $\{lm_p^{v_1}\}$ for v_i
  the intervals of adjacent landmarks $\{[lm_p^{v_i}, lm_{p+1}^{v_i}]\}$ are added to each leaf node of p
  for each leaf in p **do**
    calculate $Proj_{v_j}(R_r), Proj_{v_j}(R_w)$, for $\forall v_j \in V_{obs}$
    if detectable at one partition **then**
      label the node DD
    else if $|lm_p^{v_i}, lm_{p+1}^{v_i}| \geq 2 * D$ **then**
      $mid = (lm_p^{v_i} + lm_{p+1}^{v_i})/2$
      add $[lm_p^{v_i}, mid], [mid, lm_{p+1}^{v_i}]$ to p
    else
      label the leaf ND
  if at least one leaf is labeled DD **then**
    break loop
merge ND partitions
get qualitative relations for each partition

5 EXAMPLE

A simple Air Conditioning system has 3 components, Blower, Distribution and Cabin. Figure. 3 shows the model in Matlab/Simulink environment. p_i is pressure, f_i is airflow rate, E is the electric power

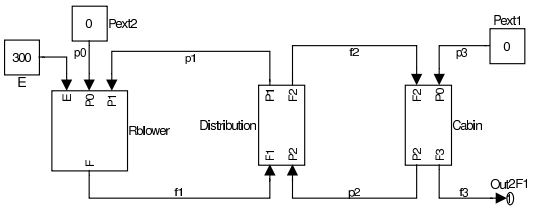

Figure 3. A Simplified AC System with 3 Components.

driving the blower. The flow rate and pressure inside the system increase when the blower begins to work; they reach a stable point if

E is unchanged. Assume the fault is a leak in the cabin, the fault is simulated as the increase of volume. The transient procedure of the fault scenario is slower than the normal case, but both reach the same stable point. It is a dynamic fault. We consider the detectability of the fault. Assume any pressure is measurable if necessary. Pressure is chosen because normally pressure is easier to measure than flow rate. Since $p0$ and $p3$ are equal to outside air pressure, they have no influence on detectability. The set $V_o = \{p1, p2, dp2\}$, where $dp2$ is the derivative of $p2$ to describe system dynamics, is the variables crucial for detectability. $dp2$ is so called ***pseudo variable*** in [22]. It is a special treatment for dynamic faults when using state-based diagnosis technique. We draw the projections of $p1$-$p2$ and $p1$-$dp2$ for both right and faulty scenarios and find that the two modes are close to each other at the projection of $p1$-$p2$, but no so close at the projection of $p1$-$dp2$. Obviously $p1$ can be chosen as the character variable, and the discrepancy of the two modes can be observed at $dp2$. Figure. 4 shows the results of using algorithm 1 on the relation of $p1$-$dp2$. Except the rightmost and leftmost intervals of $p1$, all other intervals are Deterministic Diagnosable regions. The landmarks of other variables are computed accordingly not described in this paper). Comparing with the algorithms in [22], algorithm 1 gives less landmarks, thus the qualitative model is more parsimonious.

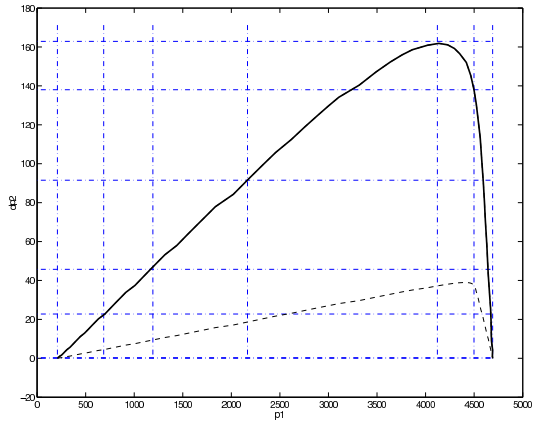

Figure 4. the relation of $p1$-$dp2$ and the partitions. The solid curve is the right behavior and the dotted curve is the faulty behavior. The landmarks are the horizontal and vertical lines.

6 CONCLUSION

The initial landmarks resulting from $\delta-$monotonicity analysis are minimal, which makes the resulting model parsimonious. The parsimony also relies on the selection of character variables, which is a topic for further study. Our monotonicity analysis can also be used for constructing other qualitative models since it is computationally efficient while being reasonably simple and robust.

REFERENCES

[1] P. Agarwal, S. Har-Peled, and Y. Wang, 'Near-linear time approximation algorithms for curve simplification'. 2003.

[2] Martin Brooks, 'Approximation complexity for piecewise monotone functions and real data', *Computers and Mathematics with Applications*, **27**(8), (1994).

[3] Martin Brooks, 'Monotone simplification in higher dimensions'. unpublished, 2004.

[4] H. Carr, J. Snoeyink, and U. Axen, 'Computing contour trees in all dimensions', in *ACM-SIAM Symposium on Discrete Algorithms*, pp. 918–926, (2000).

[5] Yi-Jean Chaing and Xiang Lu, 'Simple and optimal output-sensitive computation of contour trees', Technical Report TR-CIS-2003-02, Polytechnic University, (2003).

[6] L. Console, G. Correndo, and C. Picardi, 'Deriving qualitative deviations from matlab models', in *Proc. of 14th Int. Workshop on Principles of Diagnosis*, pp. 87–92, (2003).

[7] J.M. Díaz-Báñez, F. Gómez, and F.Hurtado, 'Approximation of point sets by 1-corner polygonal chains', *INFORMS Journal on Computing*, **12**(4), 317–323, (2000).

[8] D. Dressler and P. Struss, 'A toolbox integrating model-based diagnosability analysis and automated generation of diagnostics', in *Proc. of 14th Int. Workshop on Principles of Diagnosis*, pp. 99–104, (2003).

[9] K. Forbus, 'Qualitative process theory', *Artificial Intelligence*, **24**, 85–168, (1984).

[10] S.L. Hakimi and E.F. Schmeichel, 'Fitting polygonal functions to a set of points in the plane', *Graphical Models and Image Processing*, **53**, 132–136, (1991).

[11] T. He, L. Hong, A. Varshney, and S. Wang, 'Controlled topology simplification', *IEEE Transactions on Vizualization and Computer Graphics*, **2**(2), 171–184, (June 1996).

[12] Y. Iwasaki and H. Simon, 'Causality and model abstraction', *Artificial Intelligence*, **61**, 143–194, (1992).

[13] Eamonn J. Keogh, Selina Chu, David Hart, and Michael J. Pazzani, 'An online algorithm for segmenting time series', in *ICDM*, pp. 289–296, (2001).

[14] B.J. Kuipers, 'Qualitative simulation', *Artificial Intelligence*, **29**, 289–338, (1986).

[15] S. Morse, 'Concepts of use in computer map processing', *Communications of the ACM*, **12**(3), 147–152, (March 1969).

[16] P. E. Pfeiffer, *Concepts of Probability Theory*, Dover, 2nd edition edn., 1998.

[17] M. Sachenbacher, *Automated qualitative abstraction and its application to automotive systems*, Ph.D. dissertation, Inst. Informatik, TU Munich, 2001.

[18] P. Struss, 'Automated abstraction of numerical simulation models - theory and practical experience', in *Proceedings of 16th International Workshop on Qualitative Reasoning*, pp. 161–168, (2002).

[19] D. Suc and I. Bratko, 'Induction of qualitative tree', in *Proc. of European Conference of Machine Learning in 2001 (LNCS 2167)*, pp. 442–453. Springer, (2001).

[20] M. van Kreveld, R. van Oostrum, C. Bajaj, V. Pascucci, and D. Schikore, 'Contour trees and small seed sets for isosurface traversal', in *ACM Sympos. Comput. Geom.*, pp. 212–220, (1997).

[21] B. Williams, 'Interaction-based invention: designing devices from first principles', in *Recent Advances in Qualitative Physics*, 413–433, MIT Press, Cambridge, MA, (1992).

[22] Y. Yan, 'Qualitative model abstraction for diagnosis', in *Proc. of 17th Int. Workshop on Qualitative Reasoning*, pp. 171–179, (2003).