Semi-Quantitative System Identification

Herbert Kay*

Bernhard Rinner
Institute for Technical Informatics
Technical University Graz
A-8010 Graz, AUSTRIA
b.rinner@computer.org

Benjamin Kuipers
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712, U.S.A.
kuipers@cs.utexas.edu

February 10, 2000

Abstract

System identification takes a space of possible models and a stream of observational data of a physical system, and attempts to identify the element of the model space that best describes the observed system. In traditional approaches, the model space is specified by a parameterized differential equation, and identification selects numerical parameter values so that simulation of the model best matches the observations. We present SQUID, a method for system identification in which the space of potential models is defined by a semi-quantitative differential equation (SQDE): qualitative and monotonic function constraints as well as numerical intervals and functional envelopes bound the set of possible models. The simulator SQSIM predicts semi-quantitative behavior descriptions from the SQDE. Identification takes place by describing the observation stream in similar semi-quantitative terms and intersecting the two descriptions to derive narrower bounds on the model space. Refinement is done by refuting impossible or implausible subsets of the model space. SQUID therefore has strengths, particularly robustness and expressive power for incomplete knowledge, that complement the properties of traditional system identification methods. We also present detailed examples, evaluation, and analysis of SQUID.

keywords: qualitative reasoning; system identification; qualitative simulation; monitoring; diagnosis; imprecise models

*See note at end of paper.

1 Introduction

System identification, in its simplest textbook form, starts with a black box with several measurable inputs and outputs. The goal is to build a model of the mechanism, typically an electrical circuit, inside the box. The first step, called *structural identification*, involves experimenting with the inputs, observing the outputs, to determine the qualitative properties of the mechanism. The result is one or more imprecisely specified models; that is, descriptions of spaces of precise models. In the second step, called *parameter identification*, for each model input-output data is collected and analyzed to converge (in the limit) to the precise model that best fits the data.

In traditional methods for system identification [27] and monitoring [17] the imprecise model, or model space, is represented by a parameterized ordinary differential equation. Model refinement is done by collecting and analyzing input-output data to estimate numerical values for the parameters. As more data becomes available, the previous best estimate may be adjusted.

We present an alternate approach (called SQUID, for Semi-QUantitative system IDentification) to the refinement of imprecise models using observations, exploiting the strengths of qualitative and semi-quantitative representations for incomplete knowledge of dynamical systems. An imprecise model defines a space of precisely specified models, and embodies the hypothesis that the desired model lies within that space. When new information becomes available, rather than estimating the best-fitting precise model, SQUID refines the model space by pruning off those portions that are inconsistent with the new information, preserving the hypothesis that the desired model lies within the space. If the model space becomes empty, this hypothesis is refuted, so the correct model must be a refinement of some other imprecise model. Clearly, this approach to system identification is highly relevant to tasks such as monitoring and diagnosis.

SQUID is based on the QSIM representation for qualitative models and algorithm for qualitative simulation [24] and on its semi-quantitative extensions: Q2, Q3 [2] and SQSIM [18]. The model space is represented by a *semi-quantitative differential equation (SQDE)*, which defines a set of ordinary differential equations consistent with qualitatively described *landmark values* and *monotonic function constraints*. The semi-quantitative description includes real-valued bounds associated with the landmark values, and real-valued functional envelopes associated with the monotonic function constraints. The model space for the SQDE is the product of the model spaces for the individual landmark values and monotonic function constraints. Model refinement unifies the observation stream with the predictions from the SQDE, shrinking the bounds and envelopes. If any bound shrinks to the empty set, the product is empty, and the entire model space is refuted.

This paper focuses on the representation of imprecise models and their refinement with information from the observation stream. Model creation is addressed by research on building qualitative models [9, 31, 10, 15, 34] and on model-based diagnosis [11, 30, 29, 28]. SQUID fits within the MIMIC approach to monitoring

[12, 13, 14] and significantly extends its model-refinement capabilities.

SQUID provides the following advantages over traditional methods for system identification.

- SQUID is conservative, since model refinement is based on refutation, eliminating only portions of the model space that are provably inconsistent. When several qualitatively distinct alternatives remain consistent with the observations, SQUID preserves them. Traditional methods, representing the refined hypothesis as a single best-fitting set of numerical values for model-space parameters, can express only a single possibility.

  This difference is particularly important since many systems in continuous operation only demonstrate a small portion of their dynamic behavior over a given observation period, so it is easy to converge too quickly on too restrictive a model.

- SQUID clearly distinguishes between the cases where (a) a new observation is consistent with the model but provides no new information; (b) the new observation provides new information, further restricting the current model space; and (c) the new observation provides new information that reduces the current model space to the empty set, refuting the hypothesis. The traditional approach, with a single best-fitting precise hypothesis, does not distinguish between these alternatives.

- SQUID is highly expressive of states of incomplete knowledge. In fault model creation or black-box system identification, it may be easy to determine that two parameters are related monotonically, but difficult to determine the functional form. Qualitative models can express this state of knowledge and converge on a more precise characterization as more data becomes available. Traditional methods must commit to a functional form at the beginning.

By relying on weaker assumptions, by refining the model space conservatively through refutation, and by being more expressive of states of partial knowledge, SQUID provides an alternate approach to system identification that complements traditional methods and helps make them more widely and robustly applicable.

The rest of the paper is organized as follows. Section 2 briefly describes the QSIM framework for qualitative and semi-quantitative representation and simulation of imprecise models. Section 3 describes SQUID in detail: how the data in an observation stream is described by semi-quantitative *trends*; how observed trends are mapped onto, and intersected with, predicted semi-quantitative *behaviors*; how the refined behavior description is propagated back to refine the bounds and envelopes of the semi-quantitative model; and how to address the problem of temporal uncertainty in the correspondence between observation and prediction. Section 4 describes a set of experimental evaluations: assessing the effect of amount and quality of observations; assessing the effect of single and multiple sources of uncertainty; assessing the effect

of observability of variables; assessing the impact of temporal uncertainty; and demonstrating the application of SQUID to the monitoring task. Section 5 discusses related work, and section 6 provides a summary and discussion of future work.

2 Representing and Simulating Imprecise Models

When describing an uncertain system, it is helpful to separate the information that is precisely known from that which is not, so that unambiguous inferences can be made from the precise information. Often, there will be precise information about the structural and qualitative properties of the model, while the numerical information will be imprecise. SQUID makes use of this distinction by using a multi-level representation based on the QSIM [24] representation for ODE systems. In this section, we therefore briefly summarize the QSIM framework for representing and simulating imprecise models.

Figure 1 shows this multi-level representation demonstrated on a simple first-order tank system.

- *The structural level (structural differential equations – SDE)*
  At this level, we describe the form of the ODE system in terms of the state variables and the constraints that link them. Constraints are described as arithmetic operators and functional relationships. The structural level provides the backbone of the modeling process.

- *The qualitative level (qualitative differential equations – QDE)*
  The qualitative level adds information about each model variable by breaking its domain into an ordered list of *landmarks* that represent important values of the variable, i.e., its *quantity space (qspace)*. In Figure 1, the qspace of variable A is given by the symbols 0 and FULL and the qspace of variable c is given by the symbols 0 and IF, respectively. The QDE also adds information about the shape of the functional constraints, e.g., monotonic (M^+, M^-) or U-shaped (U^+, U^-).

- *The semi-quantitative level (semi-quantitative differential equations – SQDE)*
  At this level, we record the uncertainty in the model. We represent parametric imprecision with numerical intervals that bound the landmark values. We represent functional imprecision by defining *static envelopes* within which the functional constraint must lie.

At each level, we further restrict the model space so that it eventually contains only a single ODE. By using simulation techniques targeted for particular levels, we can thus utilize this information in a variety of ways.

The SQSIM simulator [18] generates semi-quantitative (SQ) behaviors from an SQDE and an initial condition by using the QSIM [22, 23], Q2 [25], and NSIM [19] simulators, respectively. Thus, an SQ behavior consists of three components (see Figure 2):

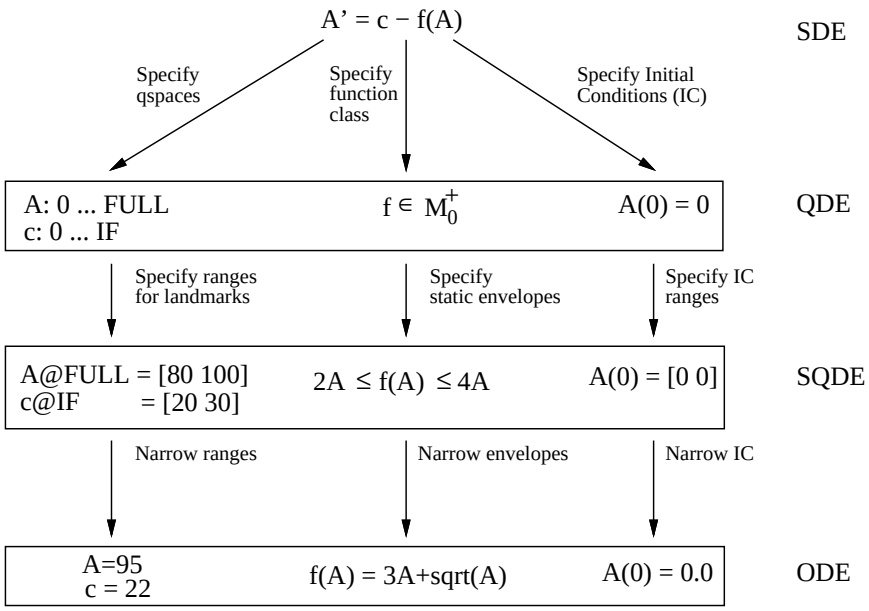


Figure 1: Multi-level model of an imprecisely-known system based on the QSIM representation. At each level, the representation entails a space of ODEs. At the top is the purely structural SDE. The QDE adds qualitative information to the SDE, and the SQDE adds imprecise numerical information to the QDE. As we move downward we specify more information, thus reducing the size of the model space.

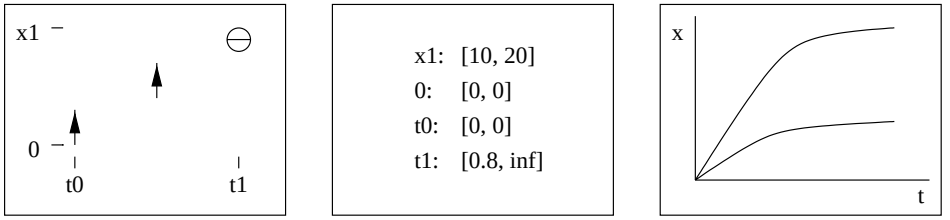


Figure 2: The SQ behavior description. SQSIM derives behavioral components at the qualitative, event and dynamic envelope level.

- The *qualitative description* is generated by QSIM and represents magnitudes and derivatives of the trajectory at time points and intervals between time points. The magnitude is expressed by a landmark or the interval between two landmarks of the variable's qspace. The symbols $\uparrow$, $\downarrow$ and $\ominus$ express the derivative's sign (qdir) of the trajectory. QSIM requires the QDE as input and produces the qualitative description by generating qspace subsets of all variables consistent with continuity conditions and the constraints at each time point and interval.

- The *semi-quantitative events* are generated by Q2. They describe the uncertainty about the value of instantaneous events such as $x(t_1) = x_1$ by providing interval bounds on t_1 and x_1. Q2 requires the SQDE as input and generates the semi-quantitative event description by propagating interval bounds among model parameters at time points and by applying the Mean Value Theorem to propagate interval bounds over time intervals.

- The *dynamic envelopes* define the overall bounds of the trajectory of the system with a pair of functions that bound all trajectories of the system. Dynamic envelopes are generated by the NSIM simulator. NSIM requires also the SQDE as input but transforms these semi-quantitative differential equations into an extremal system of ordinary differential equations (ODE). NSIM then numerically solves this extremal system; the solutions correspond to the dynamic envelopes.

The resulting SQ behaviors carry the guarantee that all real behaviors of the ODEs covered by the SQDE are covered by the SQ behavior set generated by SQSIM. This guaranteed coverage property is essential to SQUID's refinement operator.

While SQUID was conceived and implemented within the QSIM framework, it could in principle be implemented within another representational framework, as long as it provides: (a) a representation that is highly expressive of states of partial knowledge; (b) a conservative inference method that maintains a (possibly probabilistic) guarantee of covering all consistent precise models; and (c) includes inference methods for excluding portions of the model space both by refining the model description (e.g. shrinking

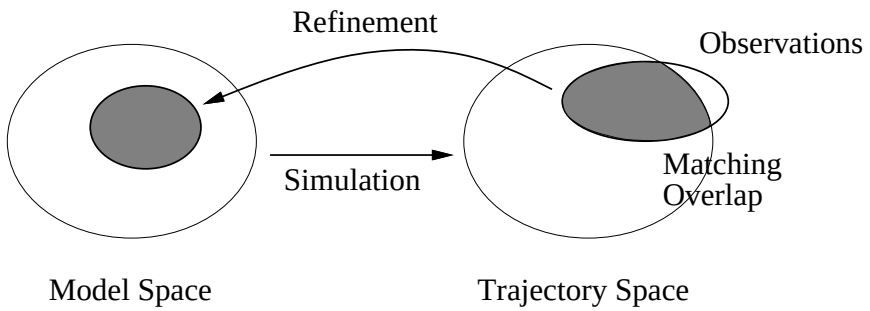


Figure 3: Refinement matches observations to the trajectory space of a corresponding model space. It then infers a smaller model space that is consistent with the smaller trajectory space. In traditional identification, this process is simplified because only one trajectory is compared at a time, so the simulation and refinement operators are easy to implement.

bounds and envelopes) and by refuting models entirely. For example, a version of SQUID built around Bayesian probability models rather than bounding intervals and envelopes would be a major contribution.

3 Refining Imprecise Models

3.1 Overview of SQUID

Refinement can be viewed as a process where measurements are matched to a model space. The portion of the model space that does not match is refuted, resulting in a more precise description of the underlying process. Because measurements and models are not directly comparable, we must match model predictions to observations and then re-map the results into the model space (see Figure 3). We define the *trajectory space* of a model space to be the set of all trajectories produced from each individual model in the model space. The mapping from model space to trajectory space is done via a simulator and the reverse mapping, i.e., refuting portions of the model space that do not match the overlap, is done by a *refinement* operator. Note that the quality of the simulation processes has a great deal of impact on refinement. In particular, we require that the simulation and refinement methods be conservative so that portions of the model and trajectory spaces are not eliminated through simulation artifacts.

In traditional identification, only one model at a time from the model space is matched against the measurements. While this approach simplifies both the simulator (which can use standard numerical methods) and the refinement (which is trivial since there is a one-to-one mapping between a precise model and its behavior), it forces one to view the model space as a collection of independent models, ignoring any natural similarity between models.

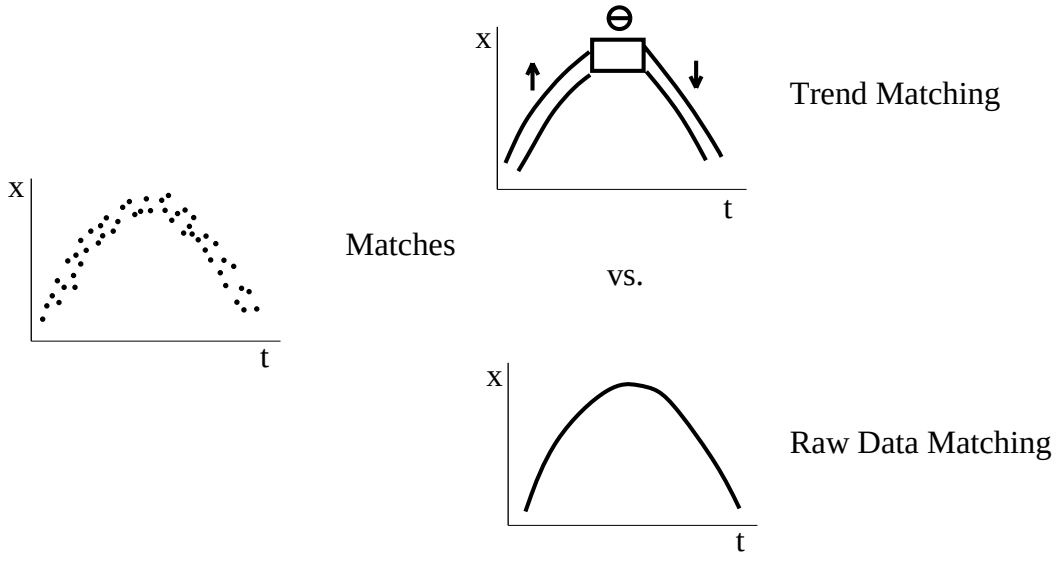


Figure 4: Trend matching uses abstract properties of the observation (monotonicity, location of events, and dynamic envelopes) to compare data to models (top). Traditional methods use specific function matches for comparison (bottom).

In contrast, SQUID views the model space as a collection of sets of models that are related by abstract properties of their associated trajectories (monotonicity, location of critical points, etc.). Matching is then performed between these abstract properties of the trajectory space and the corresponding properties of the observations, i.e., trends (see Figure 4). We call this process *trend matching* to distinguish it from the *raw data matching* of traditional methods.

Trend matching has several advantages over raw data matching:

- Abstract properties describe sets of models, thus trend matching can rule out multiple models at once.
- Since abstract properties are simpler than precise functional forms, trend matching can be cheaper than data matching.

While trend matching has many desirable characteristics, it requires more general methods for simulation and refinement since regions of the model and trajectory spaces are considered rather than single elements. SQUID adopts the SQSIM framework for describing model spaces and trajectory spaces. Model spaces are represented using the SQDE, and by applying SQSIM, their corresponding trajectory spaces can be generated.

The adoption of the SQSIM framework was motivated by its expressive power of states of incomplete knowledge and the conservative simulation methods. Thus, SQSIM provides a representation and simulator

suitable for model refinement. What remains is to define the refinement method that maps the trajectory space back to the model space. As we will show, this method can also be implemented using the Q2 portion of SQSIM. The resulting refinement method is also conservative and provides a robustness guarantee with respect to uninformative data that traditional identification cannot provide.

There are three steps to SQUID:

1. Form an SQ trend for each measured variable.
2. Map the SQ trend to the SQ behavior generated by SQSIM for the SQDE.
3. If no match is found, then the SQDE is refuted. Otherwise, refine the SQDE to rule out those portions of the model space that could not have generated the match.

These steps are discussed more fully in the following sections. As an aid to following the discussion, we will apply SQUID to the model $y'' = -9.8, \quad y(0) = 0, \quad y'(0) = v_0, \quad v_0 \in [20, 60]$ with data drawn from the same model with $v_0 = 50$ and additive Gaussian noise with variance $\sigma^2 = 5$. This model represents the effect of gravity on an object thrown into the air with an initial velocity v_0.

3.1.1 Requirements for Using SQUID

SQUID makes the following assumptions about the identification problem:

- A semi-quantitative model of the process to be identified exists in the form of an SQDE.
- Each measured variable has the following properties:
  - The measurement signal can be viewed as a "pure" signal corrupted by additive Gaussian noise of zero mean and fixed variance.
  - The measurements are sampled at a frequency fast enough such that the dynamics of the pure signal can be reconstructed.
  - The variance of the noise is known (although this value may be conservative).

3.2 Forming the Semi-Quantitative Trend

Since SQUID maps the SQSIM behavior prediction to the corresponding properties of the observation, the SQ trend consists of qualitative, event and dynamic envelope descriptions for each variable in the measurement set. The first two components are generated through a process referred to as qualitative filtering (or *binning*) which breaks the measurement stream into monotonic segments (or *bins*). The dynamic envelope is generated by fitting bounding envelopes to monotonic bins using a neural-network based estimator for monotonic functions [21, 20].

3.2.1 Qualitative Filtering (Binning)

Qualitative filtering breaks the measurement stream into monotonic regions and intervening extrema. Each region (or *bin*) has a *sign* ($\uparrow, \ominus$, or $\downarrow$) which provides a segment of the qualitative description of the trend.

The qualitative kernel function The qualitative filter operates by applying a *qualitative kernel function* to a window of fixed size that is slid across the measurement stream. We denote the window starting at index i of the measurement stream by $\mathbf{w}_i$. The kernel function $k(\mathbf{w}_i)$ returns one of three kernel values:

- $\uparrow$ if $\mathbf{w}_i$ contains a monotonically increasing segment,
- $\downarrow$ if $\mathbf{w}_i$ contains a monotonically decreasing segment, and
- $*$ if the monotonicity of $\mathbf{w}_i$ is unknown.

To determine monotonicity, the kernel function computes the slope of a linear least-squares fit to the data within the window using the formula

$$slope = \frac{\sum_i (t_i - \overline{t})(y_i - \overline{y})}{\sum_i (t_i - \overline{t})^2} \tag{1}$$

where (t_i, y_i) is the ith data-point in the window and the bar indicates the average value over the window. Because the measurement stream includes noise, it is not sufficient to use the slope directly to determine the sign returned by the kernel function. We must determine if the slope is significantly different from zero, i.e., if it falls outside a predefined confidence range. The standard deviation of the slope of the data within the window is defined as

$$\sigma = \frac{\sigma_v}{\sqrt{\sum_i (t_i - \overline{t})^2}} \tag{2}$$

where σ_v is the given standard deviation of the measurement stream corrupted by Gaussian noise. In our implementation, the sign returned by the kernel function is based on a 3.5σ confidence range which gives a 99.9% certainty that a slope outside of this range is not zero. If the slope does not fall outside this range, i.e., $|slope| \leq 3.5\sigma$, the kernel value is $*$.

The binning strategy Binning assigns signs to data-points in the measurement stream. Note that this is not straightforward since $k(\mathbf{w}_i)$ describes the slope over all of $\mathbf{w}_i$ and not just at a single point. Instead, signs can be determined only by comparing the kernel values of adjacent windows. Let $sign(i)$ be the sign assigned to measurement i and consider two adjacent windows $\mathbf{w}_i$ and $\mathbf{w}_{i+1}$. There are four cases to consider (see Figure 5):

1. If $k(\mathbf{w}_i) = k(\mathbf{w}_{i+1}) = \uparrow$ or $\downarrow$ then $sign(i+1) = k(\mathbf{w}_{i+1})$ since any extremum in $\mathbf{w}_{i+1}$ could only occur after measurement $i+1$.

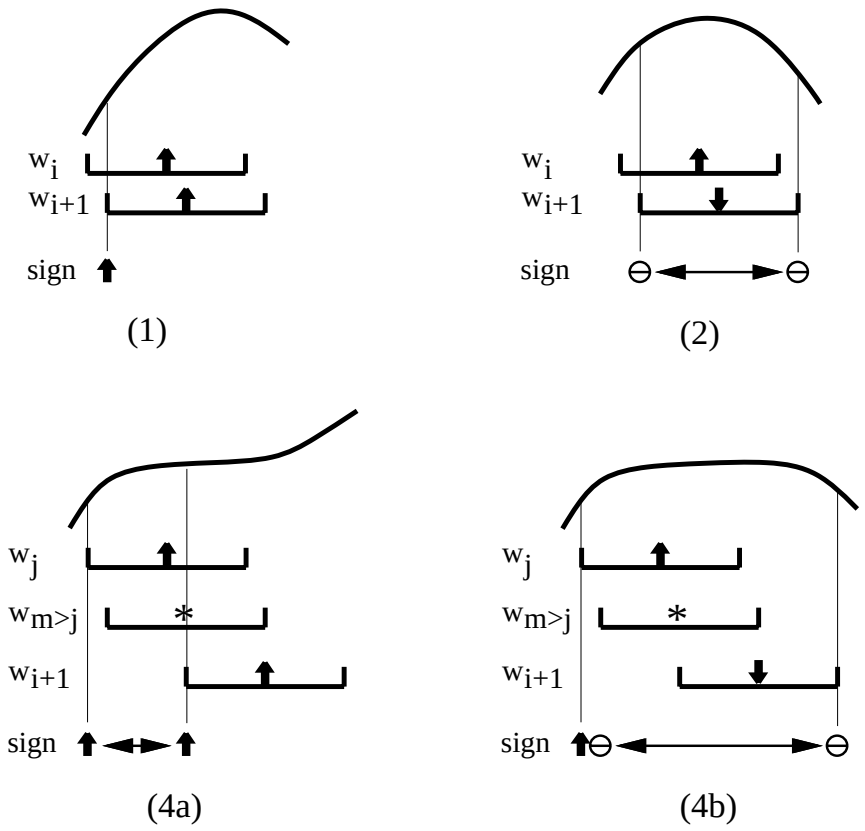


Figure 5: Determining signs (of data-points) from kernel values (of windows). Numbers in parentheses refer to the corresponding cases in the text.

2. If $k(\mathbf{w}_i) = \uparrow$ and $k(\mathbf{w}_{i+1}) = \downarrow$ (or $k(\mathbf{w}_i) = \downarrow$ and $k(\mathbf{w}_{i+1}) = \uparrow$) then there must be an extremum inside $\mathbf{w}_{i+1}$ since there is a change in slope.

3. If $k(\mathbf{w}_{i+1}) = *$ then $\mathbf{w}_{i+1}$ may or may not contain an extremum and so no sign assignment can be made.

4. If $k(\mathbf{w}_i) = *$ and $k(\mathbf{w}_{i+1}) = \uparrow$ or $\downarrow$ then a sign assignment can be made. Let $j < i$ be the last data-point that has a sign. There are two cases:

   (a) If $sign(j) = k(\mathbf{w}_{i+1})$ then all points between j and $i+1$, inclusive, must have the same sign since there has been no explicit sign change.

   (b) If $sign(j) \neq k(\mathbf{w}_{i+1})$ then there is an extremum somewhere between j and the end of $\mathbf{w}_{i+1}$.

Given this interpretation for changes in kernel values, we can construct a qualitative filtering method (Figure 6). The filter outputs a sequence of bins where each bin is a contiguous sequence of data-points with

```
1     new(ubin); new(cbin)
2     i ← 1; sign(cbin) ← ∗
3     while (i ≤ n-N+1) do
4       if (k(w_i) ≠ ∗) ∧ (sign(cbin) = ∗) then
5         sign(cbin) ← k(w_i)
6       endif
7       if k(w_i) = ∗ then
8         ubin ← ubin + data(i)
9       elseif k(w_i) = sign(cbin) then
10        cbin ← cbin + ubin + data(i)
11        ubin ← ∅
12      else
13        output(cbin)
14        new(cbin), sign(cbin) ← ⊖
15        cbin ← ubin + w_i
16        ubin ← ∅
17        output(cbin)
18        new(cbin); sign(cbin) ← k(w_i)
19        i ← i + (N-1)
20      endif
21      i ← i+1
22    end
```

Figure 6: The qualitative filtering algorithm with a window of fixed size N breaks n data-points into bins and assigns a sign ($\uparrow$, $\downarrow$ or $\ominus$) to each bin corresponding to its monotonicity.

the same sign. The implementation uses two bins to construct these monotonic regions – a current bin (or `cbin`) which holds points in the current monotonic region and an unknown bin (or `ubin`) which holds points whose monotonicity is unknown.

The procedure in Figure 6 uses a window of fixed size N to compute kernel values. Unfortunately, a single window size is insensitive to slopes below the 3.5σ threshold. By selecting a larger window size, we can reduce the standard deviation of the slope σ and hence the threshold at which a kernel function returns $\uparrow$ or $\downarrow$ (since σ depends on the window size via the summation in the denominator), but this larger window size might miss dynamics in the signal. For our needs it suffices to guarantee that we can detect *any* extremum which is not due to signal noise. Thus, we start with a small window size and only consider larger filter windows when the data suggests that they may be needed.

In our implementation, we increase window sizes whenever the filter runs into a large region of windows with kernel value of $*$. Starting with a window size of N that is selected to filter out noise, if more than $3N$ points collect in *ubin* we create a new window of size $2N$ and re-process the measurement stream starting at the first measurement in *ubin*. If the kernel value for this larger window is not $*$, then this window is used to bin the data. If the kernel value for this larger window is $*$ then the measurement stream continues to be processed by both window sizes and if the smaller one finds a kernel value other that $*$, that window is

used to bin the data and the larger window is discarded. If *ubin* increases to size $6N$, then an even larger window of size $4N$ is created and the data is filtered with it as well as the smaller windows. Whenever a non-$*$ kernel value is detected, this value is used to bin the data and all larger windows are discarded. In this way, the filter selects the appropriate window size as dictated by the measurement stream.

Binning breaks a sequence of measurements for each variable into monotonic regions and intervening extrema by a qualitative filtering algorithm. This process has several properties:

- It is conservative in that bins of sign $\ominus$ will be larger than necessary. This implies that the monotonic regions will also be conservative since some of their end-points will be contained in the adjacent $\ominus$ bins.
- Signals that include regions with differing time-scales are properly binned.
- Since binning is based purely on the measurement stream, it is in no way affected by the model space to be examined.

Figure 7 shows the results of binning data from the simulated data stream of the gravity model.

3.2.2 Fitting the Bins

Binning determines the qualitative description of the measurement stream. The next step is to determine the quantitative aspects of the data, i.e., the event and dynamic envelope descriptions. Events in the measurement stream correspond to $\ominus$ bins since they represent the precise instants at which a variable reaches an extremum. Dynamic envelopes correspond to the monotonic bins since they represent the time-intervals between the extrema. Given this correspondence, we can find events and dynamic envelopes in the trend as follows:

- For each $\ominus$ bin, the width of the event associated with the bin is determined by the beginning and ending times of the bin. The height is determined by the maximum and minimum values over the bin.
- For each monotonic bin, the dynamic envelopes are determined by two functions bounding the measurements over the time interval of the bin. These bounding functions are generated by the monotonic function estimator MSQUID [20] which takes the measurements from a monotonic region and generates bounding envelopes out to any specified confidence band.

The fitting process has the following properties:

- The event descriptions are conservative since they over-bound both the width (time) and height (value) of the event.

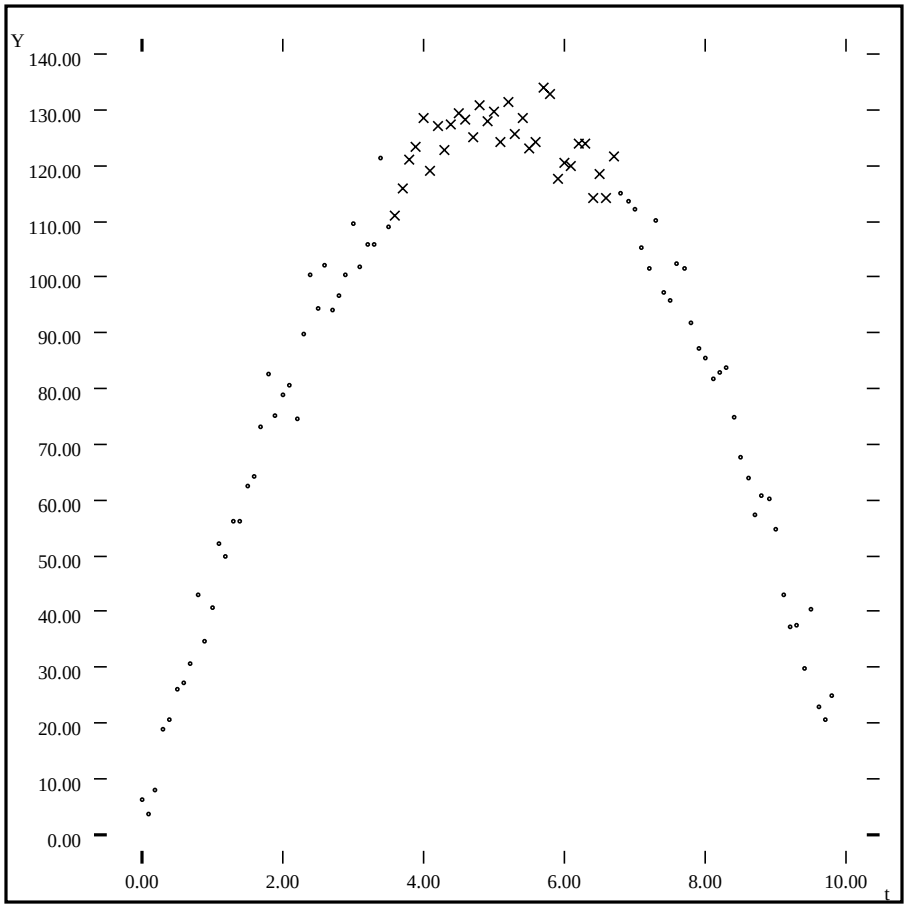


Figure 7: Binning a simulated measurement stream ($y'' = -9.8, y(0) = 0, y'(0) = 50$ with added Gaussian noise with variance $\sigma^2 = 5$). Binning finds three regions for this data – one maximum (x's) surrounded by two monotonic regions.

- The fitting process is independent of the model space. Thus, the cost of fitting can be amortized over multiple model spaces.

Figure 8 shows the result of fitting the binned data points in Figure 7. The dynamic envelopes generated by MSQUID bound the measurements within the monotonic regions with a confidence band of 3.5σ. Note that the dynamic envelopes are wider at the end of the regions because there are fewer nearby data points to constrain the envelopes.

3.3 Mapping SQ Trends to SQ Behaviors

Once the SQ trend of the measurement stream has been computed, it can be mapped to the SQ behavior generated from the SQDE by SQSIM. Since both descriptions presumably describe the same physical system, we expect that they should overlap. This overlap represents the section of the trajectory space that is consistent with both descriptions and will normally be smaller than the trajectory space defined by the SQ behavior alone.

We compare the SQ trend and SQ behavior by mapping each of their components separately. For each component, we seek to reduce the size of the SQ behavior so as to yield a smaller trajectory space for the model.

3.3.1 Qualitative Mapping

For the descriptions to be consistent, their qualitative descriptions must match. Intuitively, this match should provide a one-to-one correspondence between the bins of the trend and the qdirs of the behavior. However, there are two reasons why the match may be weaker: First, the observed trend may be a prefix of the predicted behavior. The SQ behavior is normally simulated over the time interval $[0, \infty]$. Since the measurement stream contains data over a finite time interval $[0, T]$, it is possible that it may end before some of the qualitative changes in the SQ behavior take place. Matching a prefix of the SQ behavior with the SQ trend eliminates this problem. Second, the SQ behavior may contain undetectable extrema. Because the behavioral trajectory may include qdir changes of very small magnitude, they may be undetectable in a noisy measurement stream. Thus, we relax the matching process by requiring that the trend regions appear in the proper order within the SQ behavior.

These conditions weaken qualitative mapping considerably. In particular, they permit multiple mappings between a trend and a behavior. However, because qualitative matching compares the relatively simple representations of ordered lists of the three symbols $\uparrow$, $\downarrow$, and $\ominus$, it can be performed with low computational cost. It is thus relatively easy to eliminate SQ behaviors that do not match the SQ trend at this level of description.

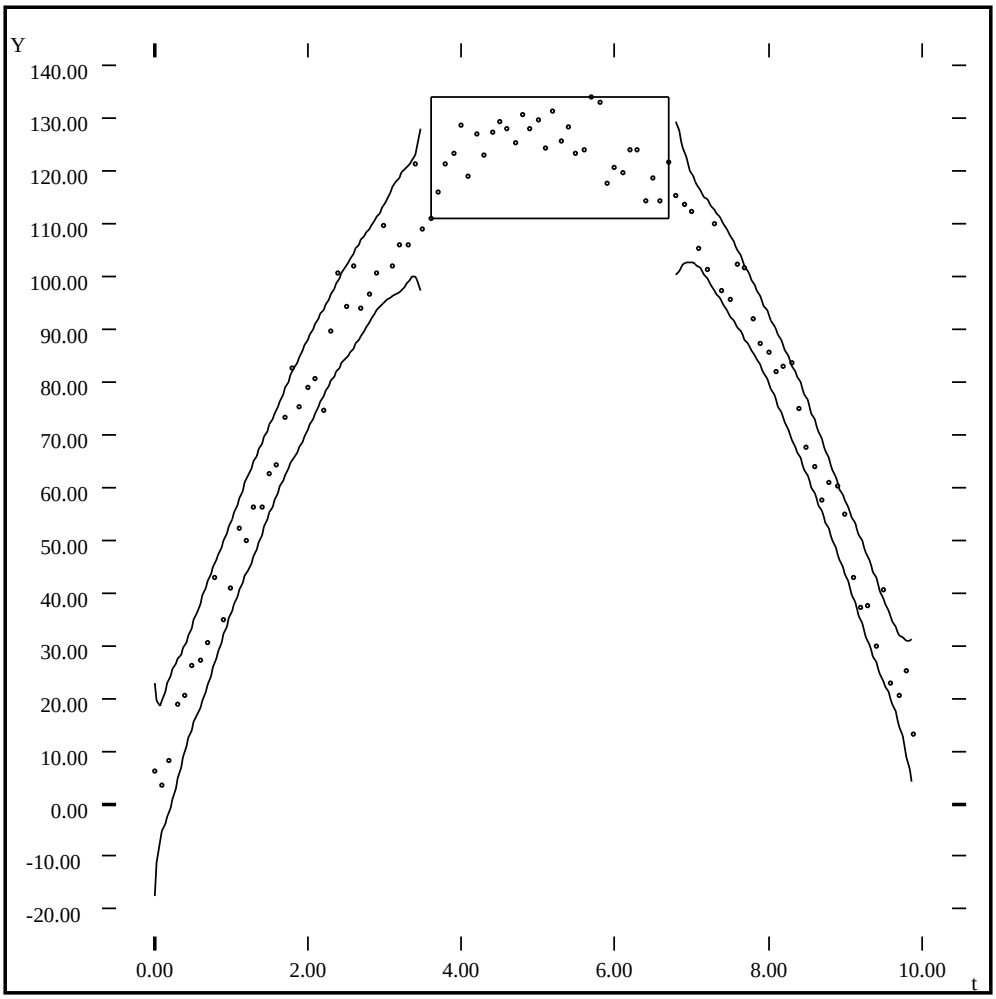


Figure 8: Fitting bins. Each $\ominus$ bin is bounded by the smallest box that includes all points in the bin. Each monotonic bin is enclosed by bounding envelopes generated by the neural network-based function estimator MSQUID.

3.3.2 Event Mapping

Event mapping ensures consistency of corresponding behavior events in the SQ trend and the SQ behavior, in the sense that their time and magnitude bounds overlap. Consistency of events is checked by asserting the event boundaries of the SQ trend to the corresponding events of the SQ behavior and propagating these boundaries to the other variables in the SQDE using Q2's interval propagation. Note that if the trend event is larger than the behavior event, it will not reduce the existing bounds on the event. If, however, it is more precise, Q2 will propagate this precision to other variables, which may refute an inconsistent mapping.

Because event mapping operates on the fixed set of symbols defined by the events in the SQ behavior, its cost is independent of the complexity of the SQ trend.

3.3.3 Dynamic Envelope Mapping

Dynamic envelope mapping ensures consistency between the dynamic envelopes for each monotonic region of the SQ trend and the dynamic envelope of the SQ behavior. The dynamic envelope for each trend region holds over the time-range of the measurements that compose the region. Consistency between the trend monotonic regions and the SQ behavior is maintained by intersecting the dynamic envelopes of each description over the time-range of the monotonic region. If the intersection is empty then the behavior is refuted.

Mapping SQ trends to SQ behaviors provides the following benefits:

- By mapping each component of the trend and behavior separately, it eliminates mismatches more efficiently. For example, a decreasing behavior can be ruled out by an increasing trend without resorting to detailed numerical analysis.

- The mapped trajectory space is conservatively reduced since qualitative mapping is conservative and the event and dynamic envelope descriptions produced by binning and fitting are conservative. Producing a conservative trajectory space aids in providing a robust refinement method.

3.4 Refining the Model

Model refinement is the process of mapping a trajectory space back to the model space that generated it. In the case of SQUID, this mapping takes an SQ behavior and determines the SQDE that covers the smallest set of ODEs that could have produced it while preserving the guarantee that no ODE is excluded unless it is genuinely impossible. Note that there are two sources of imprecision in the SQDE – variable uncertainty and static envelope uncertainty. The refinement method must therefore reduce both sources of uncertainty to refine the model.

This section begins by describing a method for refining variable uncertainty by using Q2 to derive bounds on independent variables from dependent ones. This process hinges on deriving bounds for the derivatives of state variables so that Q2 can be run on a *behavioral snapshot:* the values of state variables and their derivatives at a particular time t.

Next, a method for refining static envelopes is described. This method excludes portions of the envelope that are inconsistent with the ranges determined by the behavioral snapshot used to refine variable uncertainty.

3.4.1 Refining Variable Uncertainty

We use the Q2 interval propagator to refine variable bounds. At a time-instant, an ODE system becomes a system of algebraic equations whose left-hand sides are the instantaneous derivatives of the state variables. Q2 solves this algebraic equation system for each model variable by manipulating the SDE portion of the SQDE to form equations such that for each equation of n variables, n equations are generated with a different variable on the left-hand side. For example, for the equation

$$x' = f(c, x)$$

Q2 produces equations whose left-hand sides are x', c, and x. In particular, one of these equations is

$$c = g(x', x).$$

With this equation, Q2 can thus run the SDE "in reverse" and derive constraints on independent variables (c) from dependents (x and x')[1]. If, as we assume, our measurements reduce the bounds on the dependent variables, Q2 will be able to reduce the bounds on the independents. If, however, the measurements do not contain enough information (for example, because we do not have adequate observability, too much noise, or uninformative data) then the measured bounds will be greater than the original bounds and Q2 will be unable to reduce the model space, but no information will be lost. This is in sharp contrast to standard system identification which can be led astray by uninformative data.

As part of SQSIM, Q2 runs only at qualitative time-point states which are by definition time-instants. As part of the SQUID event-mapping process, Q2 is also run to unify the trend and behavior event descriptions. Thus, event mapping is a refinement operation. Unfortunately, since Q2 requires a time-point state, it cannot be used over time-intervals since variables over a time-interval do not represent instantaneous values. The trend dynamic envelopes, however, provide instantaneous interval bounds on values for variables and we would like to exploit this information for refinement. Thus, to extend Q2 propagation into time-intervals,

[1] Note that this property is provided since the mathematical operations that can occur in each equation are compositions of the arithmetic operators (which have clearly defined inverses) and monotonic functions (which have user-specified inverses).

we have to introduce instantaneous snapshots of the SQ behavior at any time and provide bounds for the dependent variables at those snapshots. Since bounds on state variables can be directly derived from the SQ prediction or – if available – measurements, we focus on computing bounds on derivatives at snapshots. We improve the dynamic envelope prediction for each derivative by the following method:

If we know the sign of the first and second derivatives of a state variable x over an interval $I = [t_0, t_2]$, we can infer a bound on the derivative x' by computing slopes over subranges of I. Consider the case where x is monotonically increasing over I with a decreasing second derivative (i.e., concave down). Assume we are interested in the derivative of x at $t_1 \in I$. Note that the following facts hold:

1. If the slope at t_1 is m, then for all $t < t_1$, $x'(t) \geq m$.
2. $\underline{x}(t) \leq x(t) \leq \overline{x}(t)$, where $\overline{x}(t)$ and $\underline{x}(t)$ are the upper and lower envelopes for x over I.
3. At any $t < t_1$ the maximal slope of x over $[t, t_1]$ is

$$m(t, t_1) = \frac{\overline{x}(t_1) - \underline{x}(t)}{t_1 - t}.$$

Therefore, the maximum slope at t_1 is bounded by the maximum value of $m(t, t_1)$ for $t_0 \leq t \leq t_1$ (see Figure 9). Similarly, the minimum slope at t_1 is bounded by the minimum value of $\frac{\overline{x}(t) - \underline{x}(t_2)}{t_2 - t}$ or zero. Additional equations can be derived for monotonically increasing but concave down and monotonically decreasing concave up or down functions. Note that this calculation is essentially what time-point insertion of Q3 [2] computes, except without the overhead of generating additional qualitative time-point states.

Note that we can determine bounds on the interval I from the width of the SQ behavior events since they are guaranteed not to overlap in time. This is because binning naturally breaks the trajectory into non-overlapping regions of monotonic behavior and mapping assigns these regions to the SQ behavior.

For the gravity model, variable refinement improves only the lower bound of y' at $t = 0$ over its initial value. This leads to the reduction of the initial state uncertainty from $[20, 60]$ to $[36.3, 60]$. The final prediction for y is shown in Figure 10.

Note that we can use the SQ trend of state variables to compute bounds on the derivatives as well. MSQUID computes an envelope on a nonlinear function over monotonic segments. We can estimate the envelope around the slope of the nonlinear function $\hat{y}$ by applying exactly the same method outlined in [20, 1], but with $\frac{d\hat{y}}{dt}$ in place of $\hat{y}$. Thus, a slightly modified version of MSQUID can be used to determine derivative bounds from the measurements.

3.4.2 Refining Static Envelopes

Refining a static envelope means reducing the width between the bounding functions. Assume that at some time-instant t, the ranges of x and y are $[a, b]$ and $[c, d]$. These ranges produce a box in the (x, y) plane of

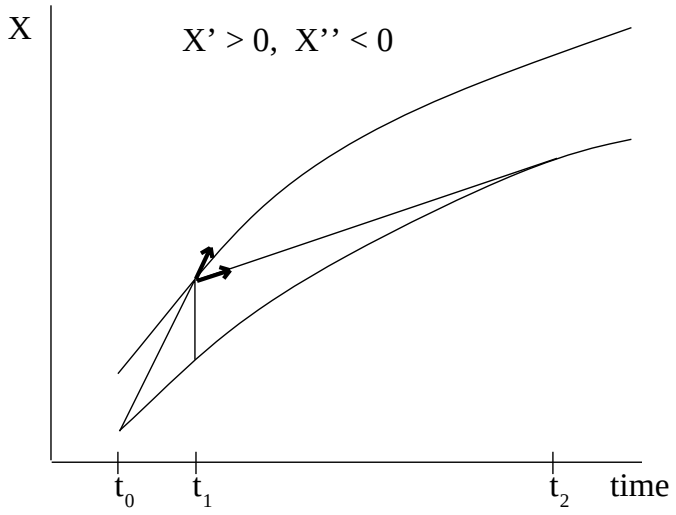


Figure 9: Computing bounds on the derivative of a state variable from its dynamic envelopes. The signs of the first and second derivatives of x provide information to determine bounds on x' from the dynamic envelope of x.

the static envelope (see Figure 11). This box is analogous to an SQ event description in that the box defines a region in which the true x and y values of the system must lie at t. If the upper-left corner of this box falls inside the static envelope for f then refinement is possible. This is because any point in the region above and to the left of this corner is unreachable for a monotonically increasing function. We may thus eliminate this region from the envelope. A similar argument rules out the region below and to the right of the lower-right corner of the envelope.

If we also have curvature information about f, we can further refine the static envelope by eliminating regions that violate the curvature assumption. For example, assume that f is concave downwards and that the upper-left corner of the range box is at (a, d). If we determine the maximal slope at this point to be m (by using the method for inferring derivatives from envelopes, for instance) then we may eliminate all portions of the envelope above the line $y = m(x - a) + d$ for all $x > a$ since any point above this line could only be reached by a path that has a slope greater than m (see Figure 11). Consideration of the minimal slope and lower-right corner of the bounding box eliminate further portions of the static envelope.

The gravity model does not include static envelopes. We will therefore demonstrate static envelope refinement on the first-order model

$$A' = 10 - f(A) \qquad A(0) = 0$$

where $f \in M^+$ with static envelope $2A \leq f(A) \leq 5A$ and the maximum value for A is represented by the landmark $FULL \in [50, 60]$. We use a data source for A computed from $A' = 10 - 3A$ corrupted with

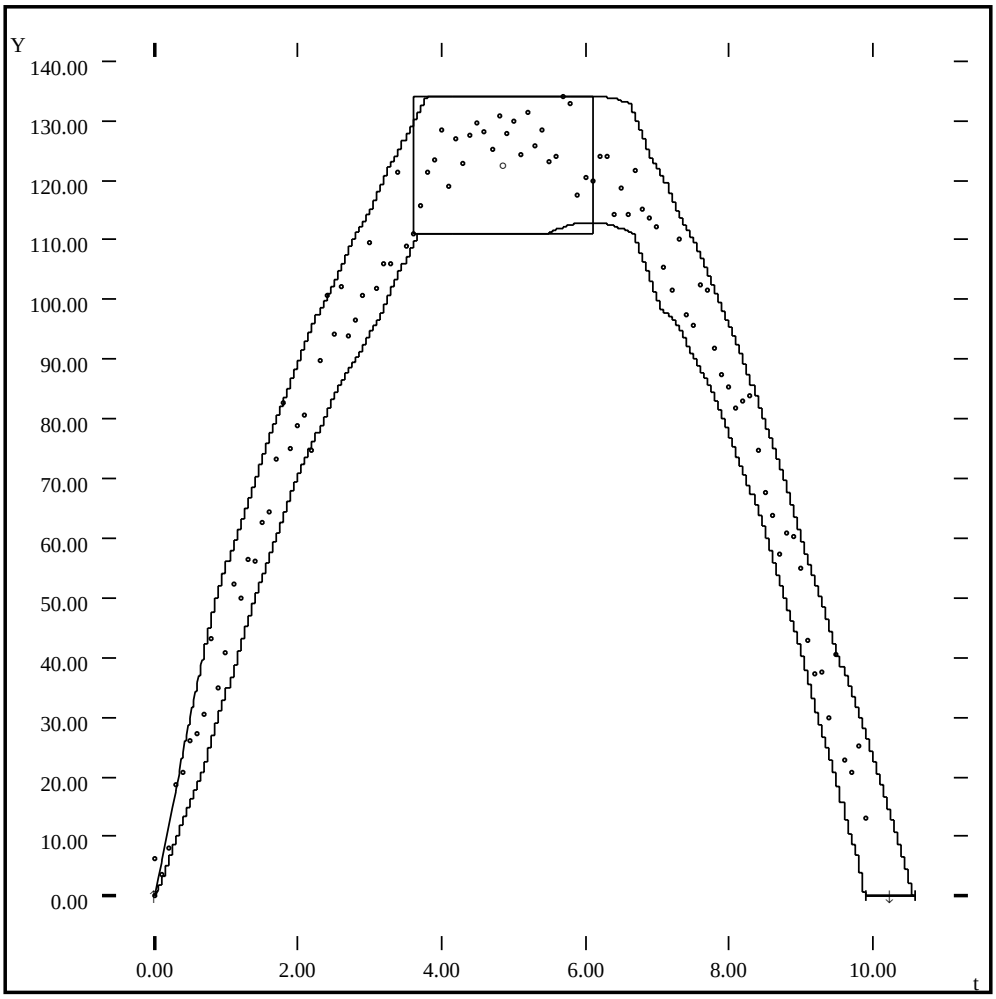


Figure 10: The predicted trajectory of the gravity model after refinement. The bound on initial velocity has been reduced from $[20, 60]$ to $[36.3, 60]$. Note that the bound for y near $t = 0$ is better than that of the trend dynamic envelope (Figure 8) since the model prediction provides greater constraining power than do the measurements at that time.

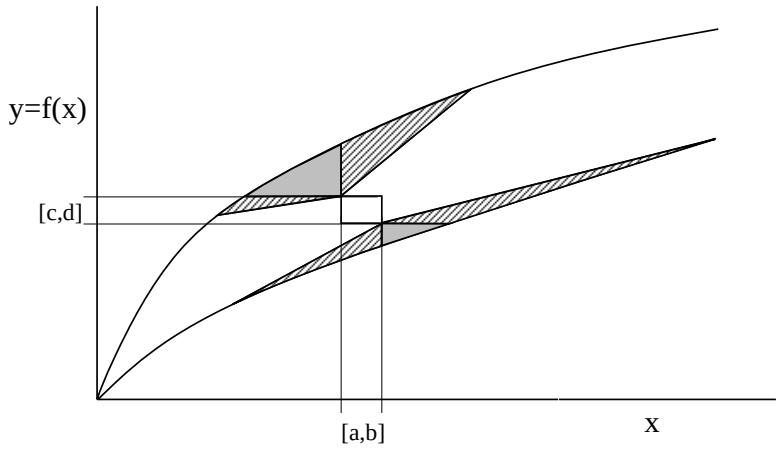


Figure 11: Refining a static envelope. Monotonicity requires that an increase in x leads to an increase in y. Thus, if the region in the box contains a point on $f(x)$, the dark-shaded regions cannot contain parts of the function. Curvature information about $f(x)$ allows a further refinement, i.e., the elimination of the light-shaded regions for a concave down function.

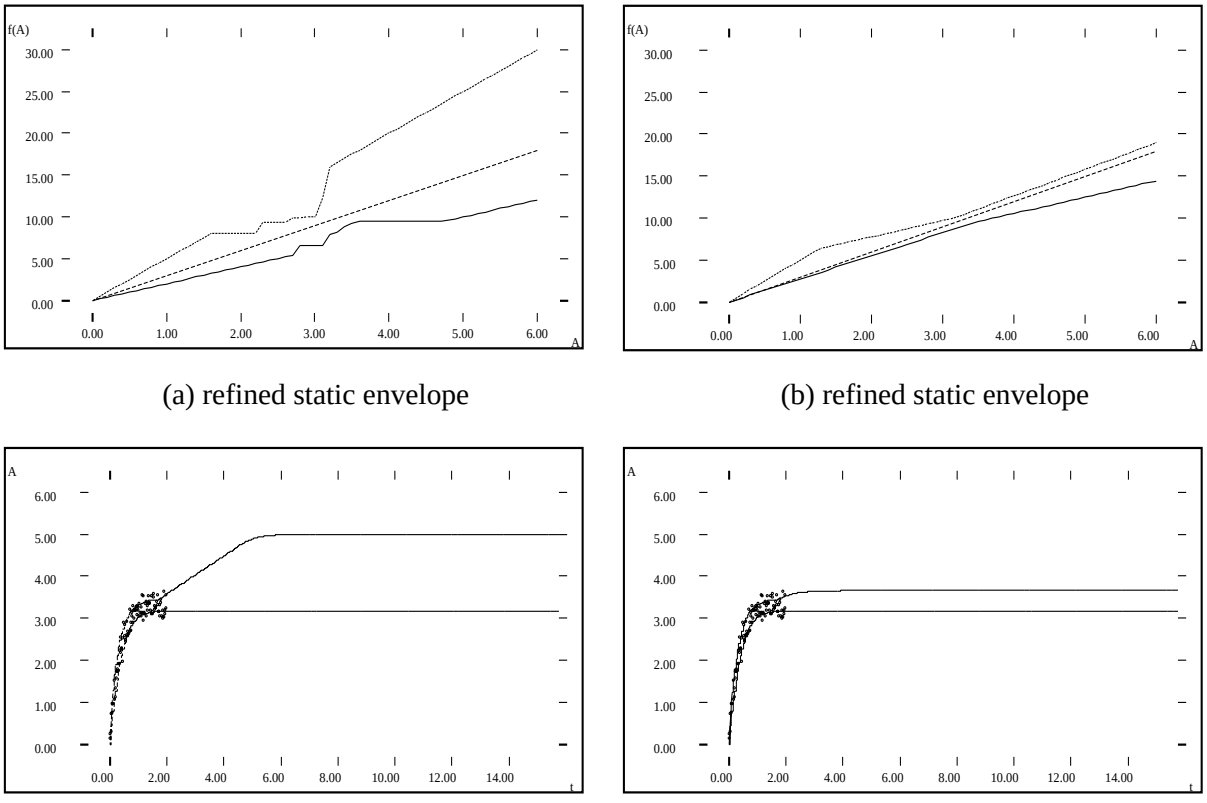
additive noise of variance 0.25 and sampled at 10 measurements per time unit. SQUID is then applied to the SQ behavior and data source. Figure 12 shows the resulting uncertainty in f (top) and their effect on the behavior prediction (bottom) with and without using curvature information. In Figure 12a, no knowledge of curvature or the range of the *FULL* landmark is given. Note that the static envelope is reduced, although the effect is very localized about the A values at the snapshot time points. If we add the further information that f is concave downward ($f''(A) < 0$) and $FULL \in [50, 60]$ and rerun SQUID on the same data, we obtain the results in Figure 12b. Notice that the static envelope is much improved. Figures 12c and 12d show the effect of the improved static envelope in the prediction. Using the additional curvature information greatly improves the predicted dynamic envelope as seen by the reduction of the upper bound from 5 to 3.6 (Figure 12d).

In the previous sections, we have described the three steps of SQUID, i.e., trend forming, trend mapping and model refinement, necessary to refine an imprecise model given a behavior prediction and uncertain observations. In the following section, we discuss the effect of an additional source of uncertainty on SQUID's refinement capability: the time uncertainty between prediction and observation.

3.5 Time Uncertainty Between SQ Trend and SQ Behavior

An inherent problem of monitoring and diagnosis applications is that the initial knowledge of a hypothesis like a fault may be very weak. More specifically, the exact starting time of the hypothesis with respect to

without curvature information **with curvature information**



(a) refined static envelope (b) refined static envelope

(c) predicted dynamic envelope (d) predicted dynamic enevlope

Figure 12: Static envelope refinement (top) and the effect on behavior prediction (bottom). Adding additional information about the curvature of the static envelope greatly reduces the uncertainty in both the refined static envelope and the predicted dynamic envelope (right). The dashed lines in the upper graphs represent the true function ($f(A) = 3A$).

the data observed may not be known. Only bounds on this instantaneous time may be specified. We call this interval on the starting time of a hypothesis H_i the *time uncertainty* $t_u = [\underline{t_u}, \overline{t_u}]$ of H_i where lower and upper bars indicate lower and upper bounds, respectively.

Time uncertainty affects the entire correspondence between the prediction and the observation. The time scales of the observed SQ trend t_o and the predicted SQ behavior t_p no longer have a precisely known relationship. The SQ trend can be shifted relative to the SQ behavior by any offset within the range of the time uncertainty ($t_p = t_o - t_u$). This variable time offset must taken into account when the overlap between the SQ trend and SQ behavior is determined.[2] Thus, trend/behavior mapping is affected by time uncertainty in the following way:

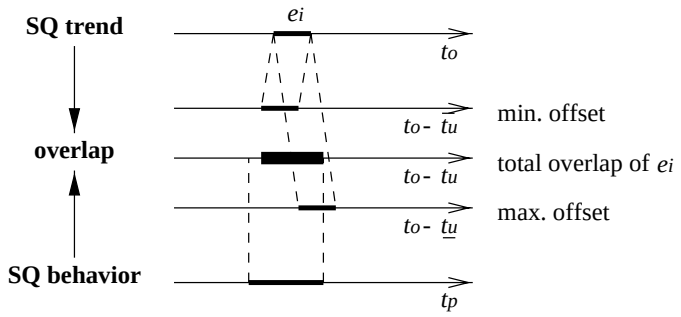
Event Mapping The time uncertainty t_u enlarges the overlap's time bound of an event e_i. To map the observed time bound $t_{obs}(e_i)$ of e_i to the trajectory space of the SQ behavior, the time uncertainty (interval) must be subtracted from $t_{obs}(e_i)$. This enlarged time bound $(t_{obs}(e_i) - t_u)$ is then intersected with the predicted time range of e_i (Figure 13(a)).

Dynamic Envelope Mapping Time uncertainty affects dynamic envelope mapping in two ways. First, the time range of a monotonic region in the trajectory space decreases because the time intervals of the adjacent events increase. Second, the magnitude overlap between the SQ trend and the SQ behavior enlarges. Magnitude bounds of the SQ behavior $X_p(t_s)$ are intersected with magnitude bounds of the SQ trend at any time-point t_s within the monotonic region. In the presence of time uncertainty, the time instant t_s in the SQ behavior corresponds to the time interval $t_s + t_u$ in the SQ trend. Thus, the trend's magnitude bound used for the overlap is given by the minimal and maximal values of the envelopes $X_o(t)$ over the time range $[t_s + \underline{t_u}, t_s + \overline{t_u}]$ (Figure 13(b)).

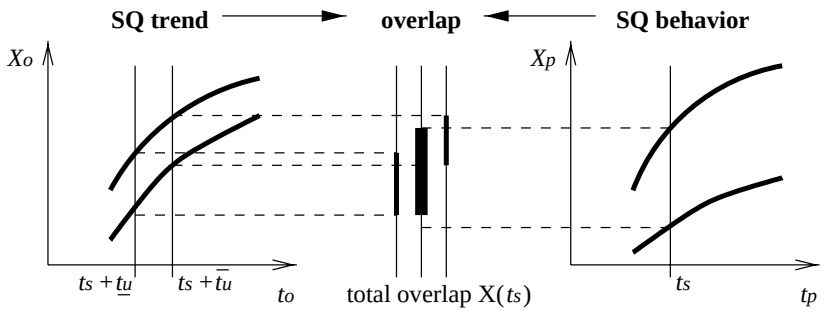
Note that for a valid mapping the intersection between SQ trend and SQ behavior must be non-empty at any time offset within the time uncertainty. This precondition can be exploited to narrow the time uncertainty before the actual trend/behavior mapping takes place [33].

Time uncertainty results in broader numerical bounds in the trajectory space and, therefore, in less effective refinements. However, the mapping process between SQ trend and SQ behavior remains conservative and no modifications are needed for the model refinement step.

[2] The overlap may be represented in the time scale of either the SQ trend or the SQ behavior. Since model refinement uses the SQ behavior as time reference, we represent the overlap in time scale of the SQ behavior.



(a) event mapping



(b) dynamic envelope mapping

Figure 13: The effect of time uncertainty t_u in trend/behavior mapping. In the presence of time uncertainty the SQ trend can be shifted from the SQ behavior by any time offset within the time uncertainty ($t_p = t_o - t_u$). Time uncertainty increases the time overlap between the SQ trend and the SQ behavior of events (a) and the magnitude overlap of dynamic envelopes (b). These broader bounds weaken the refinement process.

4 Analysis and Evaluation

4.1 The Models Used for the Experimental Evaluations

This section explores the capabilities and limitations of SQUID as demonstrated through several illustrative models. Four models are used for the experimental evaluation of SQUID:

- A single tank with constant inflow

$$A' = c - f(A)$$

  where c is a constant and f is a monotonically increasing function that is concave down. The state variable A has a landmark $FULL = [50, 60]$ which represents the maximum amount that the tank can hold. The system is simulated from $A(0) = 0$.

- A two tank cascade

$$\begin{aligned} A' &= c - f(A) \\ B' &= f(A) - g(B) \end{aligned}$$

  where c is a constant and f and g are monotonically increasing functions that are concave down. The amount in the upper tank A and the amount in the lower tank B are state variables of this model. The variable A has a landmark $FULL = [92, 98]$ and B has the landmark $FULL = [50, 60]$. The system is simulated from $A(0) = FULL$ and $B(0) = 0$.

- The gravity model

$$y'' = -9.8$$

  simulated from $y(0) = 0$, $y'(0) \in [20, 60]$. Height y and velocity y' are state variables of this model.

- A Continuously-Stirred Tank Reactor (CSTR) [18]

$$\begin{aligned} \frac{dC_A}{dt} &= \frac{C_{Ai} - C_A}{\tau} - k_0 e^{-E/T} C_A \\ \frac{dT}{dt} &= \frac{T_i - T}{\tau} - h_r k_0 e^{-E/T} C_A \end{aligned}$$

  with $C_{Ai} = 0.9$, $T_i = 340$, $\tau = 10$, $k_0 = 10000$, $E = 5000$, $h_r = -200$. Concentration C_A and temperature T are the state variables of this model.

Each model was simulated using SQSIM to produce an SQ behavior tree[3] which was used as input to SQUID along with noisy datasets.

[3] The single tank and gravity models produced only a single behavior.

4.2 The Effect of Quality and Length of Observation

A particular observation stream provides information about only a portion of the dynamics of a process. In this section, we consider the effect of decreasing noise, and of increasing length of observation, on the refinement of the model space.

We begin by examining the single-tank system with inflow $c \in [5, 10]$ and $2A \leq f(A) \leq 5A$. Figure 14 shows the result of using measurement streams of 100 points derived from the model $A' = 10 - 3A$ with noise variances of 0.25, 0.01, and 0.003 in terms of both trajectory and model uncertainty. Note that the static envelope and range on c improve with decreasing noise, thus reducing the model space. Unfortunately, the predicted dynamic envelope shows no improvement as $t \to \infty$. This is due to the effect of multiple uncertainty sources in the model, a situation that we will discuss more fully in Section 4.3.

The length of the measurement stream is another factor in informativeness since short streams may not capture all the dynamics of the underlying process. We next examine this effect on the gravity model. Figure 15 shows the resulting prediction from identifying increasingly longer measurement streams of y using the gravity model. The first prediction is generated without any data. The second prediction is generated from data that stops before the local maximum while the third makes it slightly past the maximum. Note that the prediction narrows greatly as a function of increasing data length.

4.3 The Effect of Model Uncertainty and Measurement Uncertainty on Refinement

As can be seen from the tank example in the previous section, the effect of multiple sources of uncertainty can greatly reduce the effectiveness of refinement. This is the flip-side of robustness – because interval arithmetic is conservative, its ability to reduce the model space is also conservative. In this section, we examine the reasons for why this is true.

Refinement in SQUID is affected by both measurement uncertainty and model uncertainty. Measurement uncertainty can be described by the width of the trend dynamic envelope of a measured variable. Model uncertainty can be described by the amount of uncertainty per source (i.e., the widths of the ranges of model parameters and the widths of static envelopes) and the number of uncertainty sources. In order to study the relationship between uncertainty and refinement, we make use of the single tank model

$$A' = C - Q \qquad\qquad Q = f(A)$$

where we measure the value of A'.[4] We wish to examine the conditions under which the measurement of A' permits refinement of $\underline{f}(A)$. Assume that at some time t we have a bound on A' of $[\underline{a}', \overline{a}']$. Intuitively,

[4] In this analysis, upper-case names correspond to model variables while lower-case names correspond to scalar values.

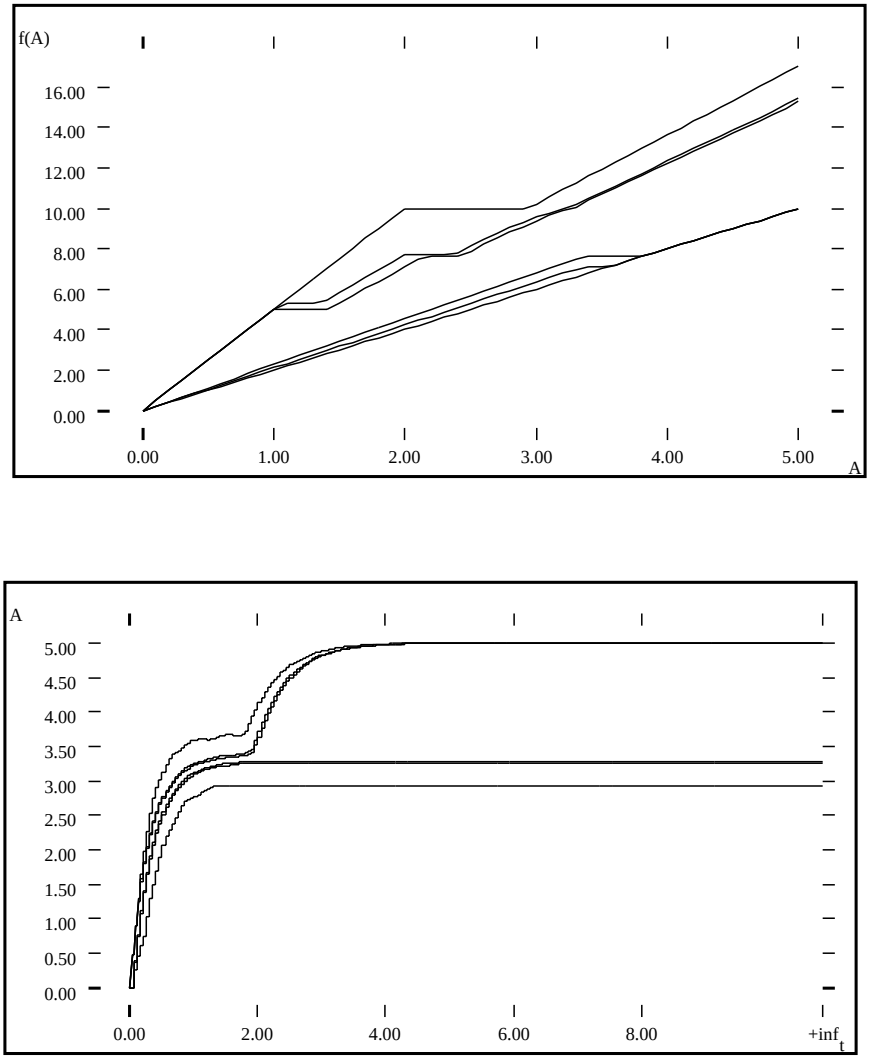
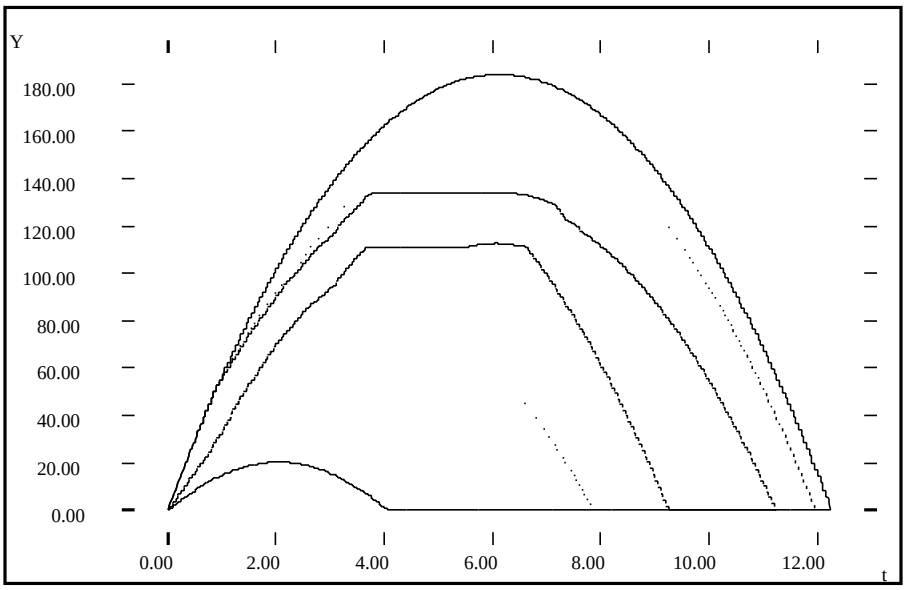


Figure 14: The effect of noise on refinement for the single-tank model. As the noise variance decreases, so does the static envelope for f (upper graph). The corresponding ranges on c are $[5.9, 10]$ for variance 0.25, $[7.4, 10]$ for variance 0.01, and $[7.9, 10]$ for variance 0.003. In all cases, a sample of 100 points over the range $t \in [0, 2]$ was used. Although the model improves with decreasing noise, the prediction from the model (lower graph) does not improve as $t \to \infty$. Note that further improvement of the dynamic envelope is possible given that it is known that the curvature of the envelope is concave downward, however this information does not lead to improving the model itself, nor does it improve the bound as $t \to \infty$.



# pts	$y'(0)$	y and t at maximum
0	$[20.0, 60.0]$	$[0, 183]$, $[2.0, 6.1]$
30	$[35.2, 60.0]$	$[89, 168]$, $[3.6, 6.1]$
80	$[35.7, 59.8]$	$[112, 134]$, $[3.6, 6.1]$

Figure 15: The effect of varying the length of the measurement stream for Y from 30 to 80 points (the sampling rate is 10 samples per unit time). The three envelopes correspond to bounds at 0 (outer), 30 (middle) and 80 (inner) measurement points. The bounds on the value for $y'(0)$ and the event $y(t) = max$ when $y' = 0$ are given in the table below the graph.

we would expect that if a model with small uncertainty can be refined by A' then a model with greater uncertainty could also be refined by the same bound.

Single-source Uncertainty Consider the single-source uncertainty SQDE derived from the above SDE where $\underline{f}(A) \leq f(A) \leq \overline{f}(A)$ and $C = c$, a precise constant. To refine the model at t, we create a snapshot-state (or use an existing one) that contains the bounds of all model variables at t. This state represents the uncertainty in $\underline{f}$ by $\underline{Q}(t)$. Q2 generates the following equations whose left-hand side contains this term:

$$\begin{aligned} \underline{Q} &= \underline{f}(\underline{A}) \\ \underline{Q} &= c - \overline{A}' \end{aligned}$$

Let us assume that the bound on $A(t) = [\underline{a}, \overline{a}]$ is such that the first of these equations improves the bound on $\underline{Q}$ so that $\underline{Q} = \underline{f}(\underline{a})$. Then in order to improve $\underline{Q}$ further it must be the case that $c - \overline{a}' > \underline{f}(\underline{a})$. Thus,

$$\overline{a}' < c - \underline{f}(\underline{a}) \tag{3}$$

defines the maximum value that an observation $[\underline{a}', \overline{a}']$ of $\overline{A}'(t)$ can attain while still permitting refinement of $\underline{f}$.

Multiple-source Uncertainty Now consider the multiple-source uncertainty SQDE where $\underline{f}(A) \leq f(A) \leq \overline{f}(A)$ and $C \in [\underline{c}, \overline{c}]$. Q2 generates the following equations for defining $\underline{Q}$:

$$\begin{aligned} \underline{Q} &= \underline{f}(\underline{A}) \\ \underline{Q} &= \underline{c} - \overline{A}' \end{aligned}$$

As before, assume that the bound on $A(t) = [\underline{a}, \overline{a}]$ is such that the first of these equations improves the bound on $\underline{Q}$ so that $\underline{Q} = \underline{f}(\underline{a})$. Then in order to improve $\underline{Q}$ further it must be the case that $\underline{c} - \overline{a}' > \underline{f}(\underline{a})$. Thus,

$$\overline{a}' < \underline{c} - \underline{f}(\underline{a}) \tag{4}$$

defines the maximum value that an observation $[\underline{a}', \overline{a}']$ of $\overline{A}'(t)$ can attain while still permitting refinement of $\underline{f}$.

Let us now assume that both the single- and multiple-source uncertainty models have the same static envelope for f and the same bounds for A at time t. Then the second terms on the right-hand sides of both Equation 3 and Equation 4 are identical. Since $\underline{c} \leq c$ we see that the value of $\overline{a}'$ that produces an improvement in $\underline{f}$ is *lower* for the multiple-source model than it is for the single source. This means that

values of $\overline{a}'$ that satisfy $\underline{c}-\underline{q} < \overline{a}' < c-\underline{q}$ will refine the single-source model but not the multiple source model. This is contrary to the intuition that models with larger uncertainty should be refined by measurements that can refine models with smaller uncertainty and suggests that our refinement operation is overly conservative.

Collapsing Multiple-source Uncertainty The problem with multiple uncertainty sources is that they are not complementary – uncertainty in one source leads to further conservatism in another. One approach to eliminating this problem is to collapse the uncertainty into a single source. For our example, consider rewriting the SDE as

$$A' = -R \qquad\qquad R = h(A)$$

where $h(A) = f(A) - C$, $h \in M^+$ and $\underline{f}(A) - \overline{c} \leq h(A) \leq \overline{f}(A) - \underline{c}$. This model replaces uncertainty in C and f with a single function h which contains all the uncertainty. For this model, the relevant Q2 equations for $\underline{R}$ are:

$$\underline{R} = -\overline{A}' \tag{5}$$

$$\underline{R} = \underline{h}(\underline{A}) \tag{6}$$

For an observation $[\underline{a}', \overline{a}']$ of $A'(t)$ to improve h, we must have

$$\underline{r} < -\overline{a}'.$$

Substituting for $\underline{r}$ using Equation 6 gives

$$\underline{h}(\underline{a}) < -\overline{a}'$$

and using the definition of h gives

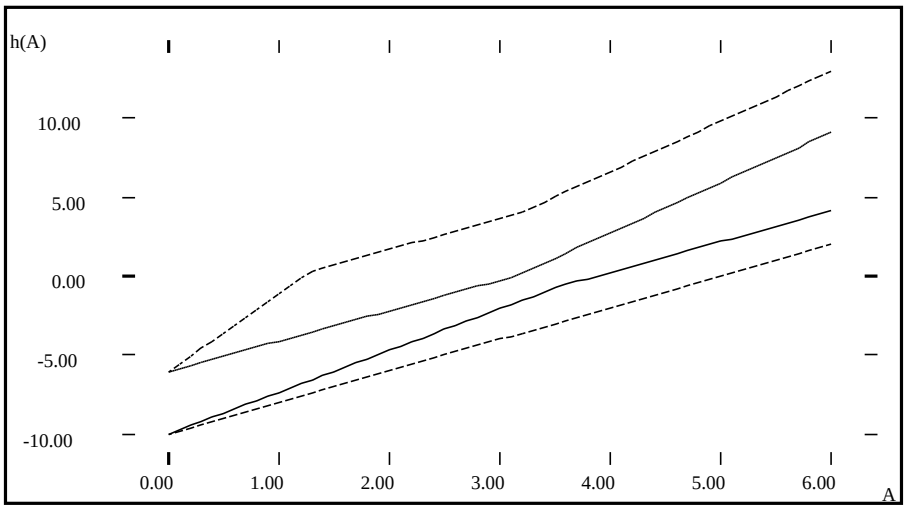
$$\underline{f}(\underline{a}) - \overline{c} < -\overline{a}'.$$

Finally, rearranging terms lead to

$$\overline{a}' < \overline{c} - \underline{f}(\underline{a}). \tag{7}$$

Let us assume that the single-, multiple-, and collapsed-uncertainty models have the same static envelope for f and the same bounds on $A(t)$. Then, since $\underline{c} \leq c \leq \overline{c}$ we see by comparing Equations 3, 4, and 7 that the collapsed uncertainty model requires the weakest bound on the upper envelope of A'. This is consistent with the expectation that greater model uncertainty requires less precision in measurement to improve the model. Figure 16 shows that this strategy leads to a greatly improved model and prediction.

The Advantages of Multiple-source Uncertainty Collapsing uncertainty sources is a useful method for improving model refinement, however, there is a cost. In the original multiple-source uncertainty model, note that the refinements of c and f are independent in that the conjunction of the statements $C \in [\underline{c}, \overline{c}]$ and



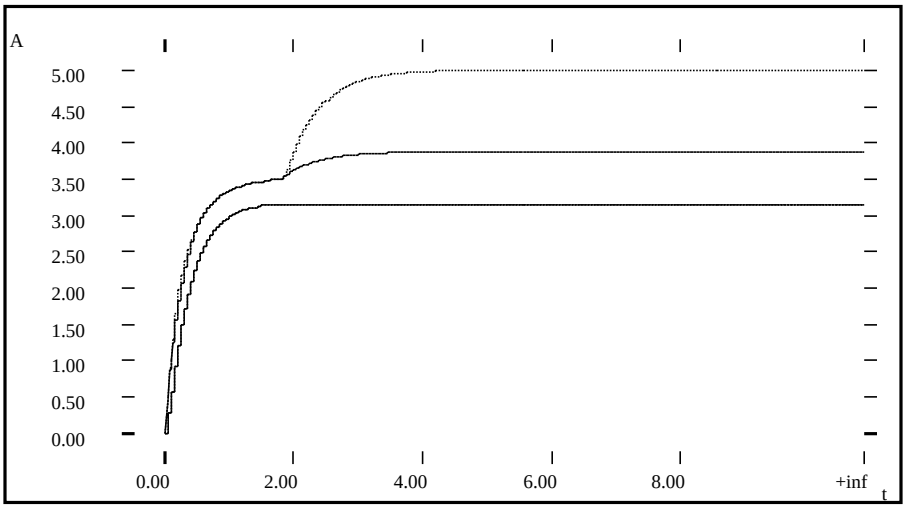


Figure 16: The effect of collapsing uncertainty sources. Using the single uncertainty source $h(A)$ rather than $f(A) - C$ leads to improvements in both the bound on $h(A)$ (upper graph) as well as the predicted trajectory whose upper equilibrium bound shrinks from ~ 5.0 to ~ 3.9 (lower graph). Dashed lines in the upper graph correspond to multiple-source uncertainty $f(A) - C$, the solid lines correspond to the refined static envelope of the collapsed multiple-source uncertainty $h(A)$.

$f(A(t)) \in [\underline{f}(A(t)), \overline{f}(A(t))]$ are true. As a result, we can extract the static envelope for f and use it in a different model with the assurance that the refined static envelope will be correct. This property is not true for traditional identification methods since *any* function that produces a satisfactory fit can be chosen. For instance, if we assume that the "true" model for $C - f(A)$ is $c_1 - f_1(A)$, the search may find the function $c_2 - f_2(A)$ where $c_2 = c_1 + d$ and $f_2(A) = f_1(A) + d$. In this case, f_2 is not a model for the true f.

By collapsing multiple-source uncertainty, we also remove the individual constraints on each individual source. This means that it is no longer possible to determine better bounds on these terms. As long as we are not interested in anything but the combination this is fine. However, if we do still care about the bounds on the individual sources, we could still include the constraints on the individual uncertainty sources. This would result in *redundant constraints* in the model which would ensure the best possible overall bound while still providing bounds on the individual sources.

4.4 The Effect of Observability

Observability is a measure of the degree to which the internals of the model can be seen. For precise models, observability determines whether the state of the system can be reconstructed from the measured variables. For imprecise models, observability also impacts the degree of refinement that can be achieved.

For our observability study, we examined the two-tank cascade with fixed inflow and ran SQUID on four cases where we measure B, A and B, B and $g(B)$, and A, B, $f(A)$, and $g(B)$. In each case, 150 measurements were generated from the model

$$\begin{aligned} A' &= 25 - 9\sqrt{A} \\ B' &= 9\sqrt{A} - 8\sqrt{B} \\ A(0) = 95 &\qquad B(0) = 0 \end{aligned}$$

with additive noise of variance 2 added to each measured variable. The SQDE is:

$$\begin{aligned} A' &= c - f(A) \\ B' &= f(A) - g(B) \end{aligned}$$

with $c \in [25, 25]$ and static envelopes for both f and g of $[1.5x, 15\sqrt{x}]$ for $x < 16$ and $[0.4(x-16)+24, 15\sqrt{x}]$ for $x \geq 16$. Table 1 shows the results of this test and Figure 17 (top) shows the static envelopes for f and g when all four variables are measured. As more variables become observable, both static and dynamic envelopes improve. Note that while $f(A)$ improves in the presence of measurements for A, the same is not true of the relationship between B and $g(B)$. This is because the differential equation $B' = f(A) - g(B)$ includes two uncertainty sources whereas $A' = c - f(A)$ includes only one. Figure 17 (bottom) presents the predicted dynamic envelopes using the refined static envelopes for f and g.

Measured variables	Envelope area ratio			
	$f(A)$	$g(B)$	$A(t)$	$B(t)$
None	1.00	1.00	1.00	1.00
B	1.00	0.97	1.00	0.13
A and B	0.51	0.97	0.28	0.13
B and $g(B)$	1.00	0.50	1.00	0.13
A, B, $f(A)$, and $g(B)$	0.28	0.50	0.16	0.12

Table 1: The effect of observability in the two-tank cascade. Each entry represents the ratio of the area of the envelope when selected measurements are made to the area with no measurements. The envelope area is defined to be the integral of the difference between the upper and lower bounds over the domain of interest. The absolute envelope areas for no measurements are 6357, 3764, 782, and 2772 when measured over $A \in [0, 100]$ for $f(A)$, $B \in [0, 70]$ for $g(B)$, and $t \in [0, 40]$ for A and B.

These results demonstrate that the best refinement is obtained when all uncertainty sources are measured. For parameters, this means directly measuring the parameter. For static envelopes, this means measuring both the domain and range of the function.

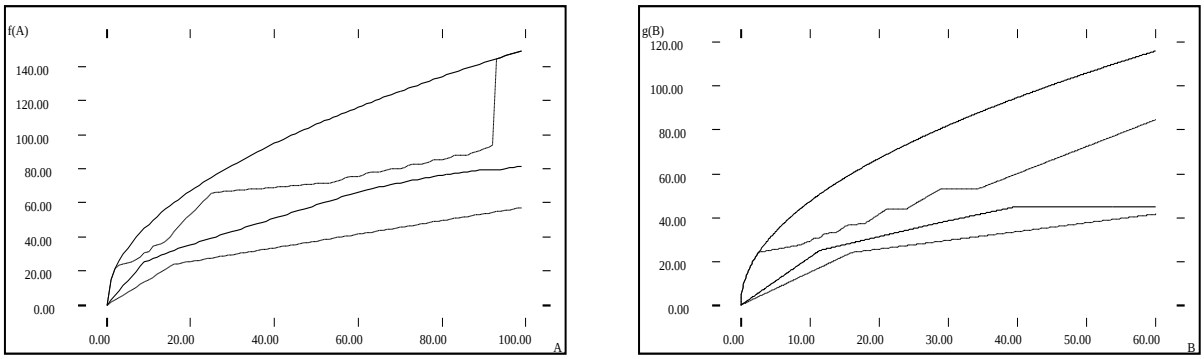
4.5 The Effect of Time Uncertainty

Time uncertainty is the time offset by which the SQ trend can be shifted from SQ behavior. It represents the uncertainty of the starting time of the simulation with respect to the observation. This section examines the effect of time uncertainty on the refinement of both static and dynamic envelopes.

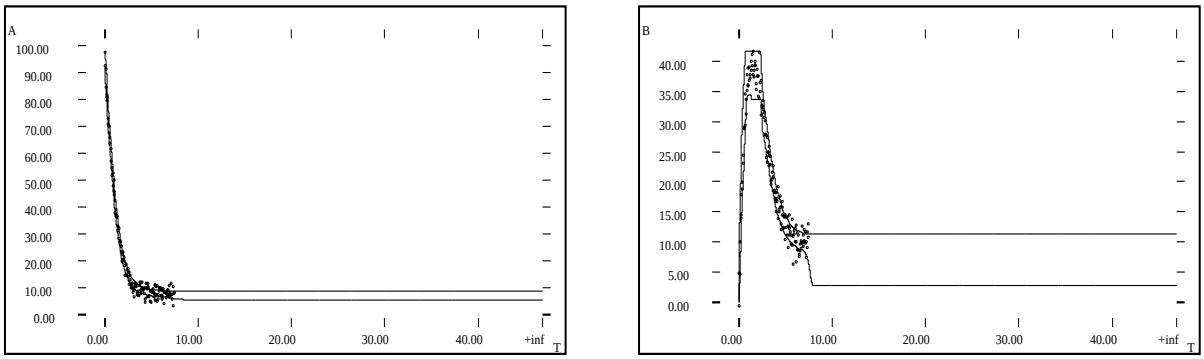
Our evaluation was based on the two-tank cascade with fixed inflow. The same SQDE and data source as in Section 4.4 were used. 150 measurements at a frequency of 20 per unit time were taken for each measured variable $A, B, f(A)$ and $g(B)$. We ran SQUID on five different cases where we increased the time uncertainty between prediction and observation from $[0, 0]$ to $[0, 2.0]$. Table 2 shows the results of this test. The refinement of both static and dynamic envelopes decreases considerably with increasing time uncertainty. Due to the wide dynamic envelopes of the prediction the improvements for $A(t)$ and $B(t)$ remain high (small envelope area ratios) even if the time uncertainty increases to $[0, 2.0]$.

4.6 SQUID as Applied to Monitoring

SQUID has been described as a method for improving monitoring applications. In this section, we examine the behavior of SQUID on such problems. We focus on two tasks of importance in monitoring – detection of model-data mismatch and detection of model drift.



refined static envelopes f and g



predicted dynamic envelopes for A and B

Figure 17: Refinement of the static envelopes f and g (top) and the predicted dynamic envelopes A and B (bottom) in the two-tank cascade when A, B, $f(A)$, and $g(B)$ are measured. The outer envelopes in the upper graphs are the initial ones provided by the SQDE. The measurements for A and B are plotted in the lower graphs.

Time uncertainty	Envelope area ratio			
	$f(A)$	$g(B)$	$A(t)$	$B(t)$
$[0, 0]$	0.28	0.50	0.16	0.13
$[0, 0.25]$	0.42	0.59	0.19	0.14
$[0, 0.50]$	0.52	0.60	0.24	0.15
$[0, 1.00]$	0.90	0.61	0.35	0.17
$[0, 2.00]$	0.91	0.65	0.42	0.20

Table 2: The effect of time uncertainty in the two-tank cascade with measurements of $A, B, f(A)$ and $g(B)$. Each entry represents the ratio of the area of the envelope when selected measurements are made to the area with no measurements.

We begin by examining the behavior of SQUID on the model-data mismatch problem. Recall that we can break monitoring into two phases – selecting an appropriate structural model and tracking the selected model. Considerable reduction of computation is possible if we can refute an incorrect structural model quickly since this reduces the numerical computation required. As an example, consider the case where we have selected a second-order model to monitor a data-stream, but in reality, our data comes from a first-order model. Figure 18 shows the result of refining the gravity model using a data-stream from the single-tank model $A' = 50 - \frac{50}{110}A$ with variance in A of 25. At up to 60 points, SQUID determines that the gravity model could correspond to the given data. Implicit is the assumption that the maximum value has not been reached. With the addition of 20 more points, however, the qualitative filter determines that the trend has no maximum before $t = 6$ and so the model is refuted.

A more subtle case of model mismatch occurs when the model is structurally correct, but the true system does not lie within the model space because the parameters and/or monotonic functions of the true system lie outside the bounds and static envelopes of the SQDE. Consider a two-tank cascade with $c \in [22.5, 27.5]$ which represents $\pm 10\%$ error on the nominal value $c = 25$ and bounds on f and g in the range $[8\sqrt{x}, 10\sqrt{x}]$. We wish to examine two separate possibilities:

1. At what point does an error in c cause refutation?

2. At what point does an error in g cause refutation?

We use a data-stream consisting of measurements for A and B with a variance of 4 from the nominal system where $c = 25$, $f(A) = 9\sqrt{A}$, and $g(B) = 8\sqrt{B}$.

In the first case, we find that the model is refuted with a 100 point data set when the true inflow is outside the range $15 \leq c \leq 35$. For values of c between 15 and 18, refinement greatly reduces the initial bounds on c while for other values, the initial ranges hold. For the second case, we vary the true $g(B) = k\sqrt{B}$ by varying

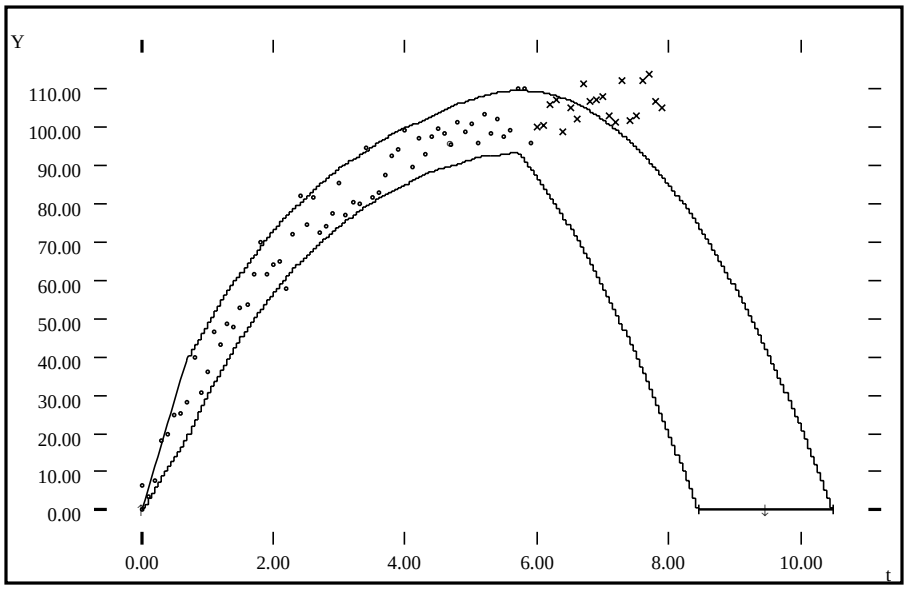


Figure 18: Structural mismatch between data and model. At up to 60 points (dots), SQUID finds a portion of the initial model space that is consistent with the measurements. After 80 points (crosses), SQUID is able to refute the model because no extrema was detected in the data before the end of the simulated maximum, i.e., event mapping resulted in an empty overlap.

k. SQUID refutes the model for $k \leq 6$.

These examples demonstrate that SQUID can detect both structural and non-structural mismatches between a model and a measurement stream. Structural mismatches are easier to detect, however, since they normally exhibit a qualitative trend description that differs from the qualitative behavior description of the model. Since only the qualitative filter is required to detect this difference, structural mismatches are normally caught before the more expensive refinement operation is performed.

As a final example, we examine the effect of drifting faults[5] in a CSTR model. We begin with the CSTR model

$$\begin{aligned} \frac{dC_A}{dt} &= \frac{C_{Ai} - C_A}{\tau} - k_0 e^{-E/T} C_A \\ \frac{dT}{dt} &= \frac{T_i - T}{\tau} - h_r k_0 e^{-E/T} C_A \end{aligned}$$

with $\pm 5\%$ in uncertainty in the nominal value $h_r = -200$ and starting at the steady-state $C_A = 0.933$ and $T = 353.36$. [18] gives a detailed discussion of this model and its behaviors when simulated from this state. We begin monitoring C_A using measurements with variance 0.0001 from a model starting at this state. At $t = 25$, we introduce a gradual fault in the inlet temperature T_i such that $T_i(t) = 340 + \frac{20}{75}(t - 25)$. As the inlet temperature shifts, SQUID refines the model as shown in Figure 19. Eventually, the data no longer matches the behavior space of the model and the model is refuted. Note that in this case refinement is not necessary since the original lower bound is sufficient to detect a discrepancy. Thus, SQUID could have simply run its mapping component, skipping refinement without any loss of diagnostic capability. Note however, a more complex analysis shows that the improvement in the lower bound has now caused more points in the region $t \in [20, 40]$ to fall outside dynamic envelope. This could be used as a signal that the model is shifting (since normally one would expect old measurements to remain in envelope). Thus, refinement could be used to detect the discrepancy earlier.

5 Related Work

In addition to traditional system identification, SQUID can also be compared to other systems developed in the AI community for trend detection, monitoring and identification.

5.1 Trend Detection in Noisy Data

Detecting a trend in the presence of noisy data is the topic of filter theory. One limitation of traditional methods is that it is often necessary to have extra information about the underlying trend such as its power spectrum to define a filter that reconstructs the original signal. Scale-space filtering [35] eliminates this need

[5] A drifting fault is caused by a gradual change of one or more system parameters.

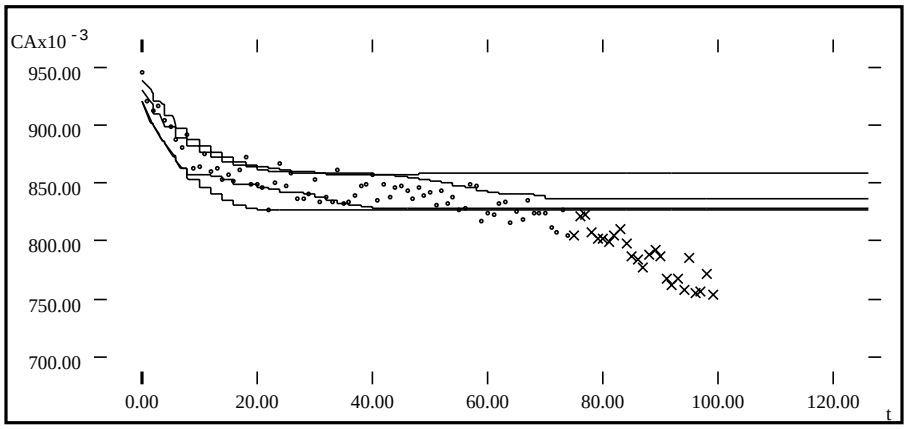


Figure 19: Refuting a drifting fault in a CSTR. The inner dynamic envelope is generated from data up to $t = 70$. The outer envelope is generated from the original SQDE. Note that the model is clearly in error from $t = 70$ onward.

by filtering the signal through a family of Gaussian filters whose filter coefficient is continuously varied across a range of values. As the coefficient increases, the signal is smoothed further until it eventually becomes flat. If we plot the inflection points of these filtered data-streams and graph them with time as the abscissa and filter coefficient (or *scale*) as the ordinate, we obtain the *scale space* of the original signal. Curves in scale-space represent the occurrence and disappearance of critical points in the signal as the scale increases. The height of the critical point on each of these curves is the scale at which the underlying inflection point is completely smoothed out. By analyzing the differences in heights of these critical points, the time-scale of the signal over different time intervals can be determined and an appropriate fitting function can be selected to reconstruct the signal at that scale.

Scale-space filtering permits the recovery of a trend with no prior knowledge. Unfortunately, it does so at the cost of filtering the signal through a theoretically infinite number of Gaussian filters. Work has been done on reducing this need [26], but it still involves multiple filtering of the data as well as pattern recognition to infer the scale-space portrait from a finite set of points.

In part as a solution to the computational expense of scale-space filtering, Cheung and Stephanopoulos developed the "triangle representation" for trend description together with an algorithm for extracting a trend from noisy data [6]. By defining a trend in terms of a primitive triangle component that captures the qualitative first and second derivatives of the trend and by describing a method for combining triangles into higher-level constructs (such as trapezoids) [7], they are able to construct a fast algorithm for constructing a scale-space and filtering a signal with it.

The SQUID binning algorithm can be seen as a variant on these trend detection techniques. Since SQUID requires only qualitative filtering (i.e., the location of regions of monotonicity in the signal), the actual filtering technique can be reduced to computing regions of constant slope in the signal. By using multiple-sized windows, it can detect slope changes at different time-scales[6].

5.2 Semi-Quantitative Monitoring and Identification

The use of semi-quantitative models for monitoring was the basis for the MIMIC system [12, 14]. MIMIC implements both the tracking and hypothesis generation phases of monitoring. It uses the SQDE as a model space representation from which an SQ behavior set is generated by (an earlier version of) SQSIM. Each behavior in the set is then matched to the measurement stream. If a model is refuted, MIMIC enters the hypothesis phase where it suggests new models to track based on the reason for the mismatch and knowledge of the structure of the device being monitored. MIMIC's strengths lie in the use of a robust prediction method (which guarantees that all possible behaviors of a an imprecise model are considered) and a hypothesis generator based on a structural model (which eliminates the need for pre-enumerating the set of possible fault models).

SQUID can be seen as an improvement to the tracking component of MIMIC. It adds a more realistic data model (MIMIC assumed that each measurement had a 100% confidence bound and that the derivative was given) together with a theory of semi-quantitative trends which make SQUID more suitable for operating on real data-streams. Also, by shifting the focus from monitoring to identification, SQUID can produce better predictions by including model refinement in the tracking process. These improvements make SQUID more efficient and robust than the tracking method of MIMIC. Of course, SQUID does not address the hypothesis generation component of MIMIC and so these methods are very much complementary.

Another semi-quantitative monitoring system is TrenDx [16]. This system also uses a semi-quantitative representation of behavior and attempts to fit data to the behavior. Unlike both SQUID and MIMIC, however, TrenDx does not use a model space representation. This has two consequences: First, since there is no model space, TrenDx cannot do refinement. Second, the user must generate the SQ behavior by hand. By sacrificing a model space representation, TrenDx simplifies the tracking component of its monitoring method. This permits more efficient methods for matching (since the user can provide customized behavior segmentation and fitting methods) at the expense of greater sophistication on the part of the user. SQUID chooses to include a model space representation together with simulation to produce an SQ behavior set. While computationally more expensive, focusing on a model allows the user to describe the structure of the process rather than exhaustively describing its behavior. Furthermore, a structural model (with a simulator)

[6] Conceivably, one could also use the SQUID binner to construct a *qualitative scale-space portrait*, although this is not necessary for SQUID.

is more flexible than a behavioral model in that one can change the structural model and then predict the consequences. This is particularly important in monitoring where faults are manifested as structural changes.

PRET [3] uses qualitative, symbolic, algebraic and geometric reasoning to automate the process of system identification. Given a set of hypotheses, observations and specifications PRET constructs an ODE model of the physical system. PRET is based on a library of traditional system identification methods and applies the reasoning techniques to select the appropriate system identification method. PRET performs a structural identification (model selection) by combining hypotheses into candidate models and a validation of those models against the observations modulo the precision inherent in the specifications. By applying standard system identification methods PRET has the same properties as traditional system identification (performed by a *human* expert). PRET focuses on helping engineers to *model* a physical system and not to *monitor* it.

Finally, another use of qualitative methods for identification is embodied in the system of Capelo, Ironi and Tentoni [5]. This system addresses the problem in traditional identification of how to select the best parametric model from a set of potential models. By using the qualitative properties of different parametric models and comparing them to the properties of the measurements, this system can eliminate from consideration those parameterizations that are inconsistent. The method does a form of qualitative trend extraction from the data which is then matched against the qualitative behavior of candidate models based on a larger set of primitives (concave, increasing, linear, etc.) but does not appear to address the problems of noise in the measurements.

6 Discussion and Conclusion

This paper has described SQUID, a new method for refining imprecise models using a stream of observations from a physical system. SQUID is based on the SQSIM framework which uses a multi-level representation for expressing and reasoning with incomplete knowledge. SQUID refines an imprecise model by a process called trend matching which compares the semi-quantitative trajectory descriptions derived by SQSIM with the corresponding properties of the observation, i.e., the semi-quantitative trend.

6.1 Comparison to Traditional System Identification

Since we are evaluating SQUID in comparison with traditional system identification, it is important to understand the situations in which one method is better than another. We can define several types of identification problems and see how each method performs on them. We look at the following:

- No functional uncertainty, small number of parameters.

  In this case, we have a precise functional form and the search space is small. This is exactly the situation that traditional identification excels at and it produces a better refined model space (see

Figure 20). SQUID can refine its initial model, however the ultimate refinement represents a much larger segment of the model space.

- No functional uncertainty, large number of parameters.
  In this case, while the functional form of the model is precise, the parameter space is large. Searching the parameter space with gradient-based methods is difficult and traditional identification may even fail to converge. Since SQUID does not search the model space, it is unaffected by the size of the parameter space and can still produce a refinement.

- Functional uncertainty.
  Since traditional identification cannot express imprecise functional models, it must approximate them using a highly parameterized model.[7] This leads to the previous case. SQUID represents imprecise functional models using bounding envelopes, which require only monotonicity within the envelopes, and no further assumption of functional form.

- Uninformative data.
  In this case, traditional identification may be led astray by reducing the model space too far because the data does not reveal the full range of the underlying dynamics. In contrast, since SQUID eliminates only inconsistent portions of a model space, uninformative data does not cause a refinement to an overly restrictive model space.

6.2 Factors Affecting Refinement

As has been discussed in Section 4, there are a number of factors that affect SQUID's ability to refine models. We summarize these factors here.

- Model uncertainty vs. measurement noise.
  The uncertainty present in the SQDE determines the maximum amount of noise tolerable in the measurements to achieve refinement. SQUID is able to make more use of a noisy dataset in case of single-source uncertainty. When multiple uncertainty sources exist, measurement bounds must be *tighter* than in the single-source case for refinement to take place. The solution to this problem is to collapse uncertainty sources. By taking the collapsing method to its limit, we can reduce each equation in the SDE to

$$x_i' = f_i(\mathbf{x})$$

[7] One can use a high-parameter neural-net estimator to replace monotonic functions. For example, translate the model $A' = c - f(A)$ to $A' = c - \hat{f}(A; \hat{w})$ where $\hat{w}$ is a vector of weights for the function estimator $\hat{f}$. This model is then used to identify the behavior of the system. Unfortunately, this approach is highly inefficient because the model space is very difficult to search.

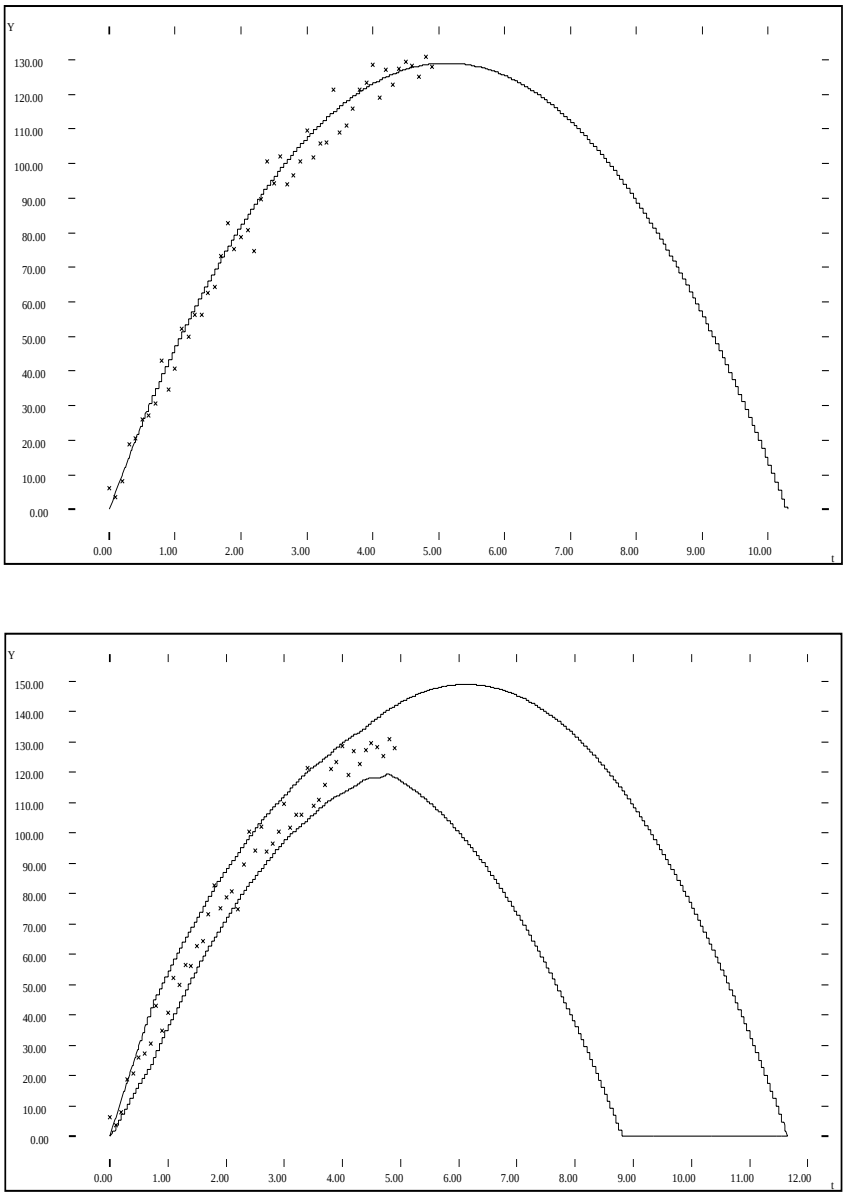


Figure 20: Comparison of the dynamic envelopes from traditional identification (top) and SQUID (bottom) for the gravity model given 50 data-points. Traditional identification excels in the case of functionally precise models with small parameter spaces (in this case, only $y'(0)$). Least square fitting results in identifying a single value for the initial velocity ($y'(0) = 50.28$ with standard deviation $\sigma = 0.12$), whereas SQUID identifies the range of possible values ($[35.5, 60]$) for which the predicted envelopes enclose the data out to a specified confidence band.

thereby creating a single uncertainty-source for each equation. The disadvantage in this approach is that individual uncertainty sources are no longer distinct but redundant constraints can be used to get the benefits of both approaches.

- The set of observable variables.
  The degree to which the trajectory space can be measured is determined by the observability of the model. In particular, SQUID does best when all imprecisely known constants and state variables as well as both sides of all monotonic functions are measured.

- Prediction imprecision.
  There are some models for which SQSIM produces numerous spurious behaviors [18]. Since SQUID relies on SQSIM to determine the trajectory space associated with an SQDE, if this trajectory space is large then SQUID must work harder to reduce the trajectory space. This can reduce both the effectiveness and efficiency of refinement. Abstraction methods [8] can collapse unnecessary distinctions between qualitative behaviors.

- Time uncertainty.
  Time uncertainty affects the mapping between the SQ trend and the SQ behavior, resulting in a larger overlap in the trajectory space. Thus, time uncertainty is an additional source of uncertainty which reduces the refinement capability of SQUID. In the future, "time alignment" methods should be developed and applied to regions of high slope to refine time uncertainty.

6.3 Summary and Future Research

To summarize, SQUID offers the following properties:

- By using the SQSIM framework, SQUID can express functional as well as parametric uncertainties. This is different from traditional identification where functional uncertainty is approximated by a highly parameterized model. Highly parameterized models complicate the search in the parameter space and are prone to converge to the wrong model. Furthermore, traditional identification may even fail to converge in this case.

- SQUID uses refutation rather than search to identify a model of a physical system. By ruling out implausible portions of the model space, SQUID is more robust in the face of uninformative data and functional model uncertainty than traditional identification methods.

- SQUID's refinement is conservative. Since all refinement steps (trend forming, trend mapping and model refinement) are conservative, SQUID guarantees that no ODE is excluded from the model space

unless it is genuinely impossible for the ODE to produce a trajectory within the refined trajectory space.

- In its current implementation, SQUID is limited to measurements that can be viewed as a superposition of Gaussian noise with fixed variance to the "pure" signal. Since SQUID uses abstract properties of the measurements it can be easily extended to other noise models of the measurements by revising the trend forming step.

Directions for future research on semi-quantitative system identification include:

- There are several areas for improving SQUID itself. First, SQUID currently operates as a batch computation over the measurement stream. We would like to make SQUID incremental, in the sense that new measurements do not require a re-computation over the entire new data set. Second, processes whose inputs vary with time are an important class of systems studied by system identification. This class of systems was excluded from our original design due to limitations in the QSIM modeling and simulation method. Recently, techniques for adding such properties to the QSIM framework have been developed [4] and should be included in SQUID. Third, adding the ability to control inputs as well as to measure outputs is necessary for SQUID to be able to solve "black box" problems. Such an extension would permit SQUID to encompass the experiment design component of system identification.

- SQUID is able to infer guaranteed bounds given uncertain hypotheses and noisy measurements. For monitoring and diagnosis, these "hard" bounds are important to distinguish whether the observation is consistent with the hypothesis. On the other hand, traditional methods using a single model with probabilistic error result in smaller but "soft" bounds. The probabilistic information of these bounds is useful in discriminating between competing hypotheses. Ideally, we would like to combine SQUID with traditional methods and benefit from both approaches.

- SQUID can be viewed as a method for tracking hypotheses and detecting discrepancies in the context of monitoring and diagnosis. To develop a complete fault diagnosis system for dynamic systems, SQUID could be combined with existing methods for automated model building [10, 32] and proposing hypotheses given weak information such as the signs of discrepancies between observations and predictions [11, 29].

Acknowledgments

This work has taken place in the Qualitative Reasoning Group at the Artificial Intelligence Laboratory, The University of Texas at Austin. Research of the Qualitative Reasoning Group is supported in part by NSF grants IRI-9504138 and CDA 9617327, by NASA grant NAG 9-898, and by the Texas Advanced Research

Program under grants no. 003658-242 and 003658-347. Bernhard Rinner is supported by the Austrian Science Fund under grant number J1429-MAT. The authors are grateful to Lyle Ungar for his helpful comments.

Note

Tragically, Dr. Herbert Kay was killed in a random act of violence on June 12, 1997. He left his wife Meg, two-year-old twin daughters Sonia and Nina, and a large group of family and friends. He left a significant body of scientific work (please see `http://www.cs.utexas.edu/users/qr/bert/`) and the unfulfilled promise of further contributions to the world, both personal and professional.

References

[1] Douglas M. Bates and Donald G. Watts. *Nonlinear Regression and Its Applications.* John Wiley and Sons, 1988.

[2] Daniel Berleant and Benjamin J. Kuipers. Qualitative and quantitative simulation: bridging the gab. *Artificial Intelligence*, 95:215–255, 1997.

[3] Elizabeth Bradley and Reinhard Stolle. Automatic Construction of Accurate Models of Physical Systems. *Annals of Mathematics of Artificial Intelligence*, 17:1–28, 1996.

[4] Giorgio Brajnik and Daniel Clancey. Temporal constraints on trajectories in qualitative simulation. In *Proceedings of the Tenth International Workshop on Qualitative Reasoning about Physical Systems*, Fallen Leaf Lake, CA, 1996.

[5] Antonio C. Capelo, Liliana Ironi, and Stefania Tentoni. The need for qualitative reasoning in automated modeling : A case study. In Yumi Iwasaki and Adam Farquhar, editors, *Qualitative Reasoning, The Tenth International Workshop*, pages 32–39, 1996. AAAI Technical Report WS-96-01.

[6] Jarvis Tat-Yin Cheung and George Stephanopoulos. Representation of Process Trends – Part I. A formal Representation framework. *Computers and Chemical Engineering*, 14(4/5):495–510, 1990.

[7] Jarvis Tat-Yin Cheung and George Stephanopoulos. Representation of Process Trends – Part II. The Problem of Scale and Qualitative Scaling. *Computers and Chemical Engineering*, 14(4/5):511–539, 1990.

[8] Daniel J. Clancy and Benjamin J. Kuipers. Qualitative Simulation as temporally extended constraint satisfaction. In *Proceedings of the 15th National Conference on Artificial Intelligence*, 1998.

[9] E. Coiera. Generating qualitative models from example behaviors. Technical Report 8901, Department of Computer Science, University of New South Wales, 1989.

[10] James Crawford, Adam Farquhar, and Benjamin Kuipers. QPC: A Compiler from Physical Models into Qualitative Differential Equations. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-90)*, pages 365–372, Boston, MA, 1990. (also in B. Faltings and P. Struss. Recent Advances in Qualitative Physics. MIT Press. 1992).

[11] Johan de Kleer and Brian C. Williams. Diagnosing Multiple Faults. *Artificial Intelligence*, 32:97–130, 1987.

[12] Daniel Dvorak and Benjamin Kuipers. Model-based monitoring of dynamic systems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1238–1243, Detroit, MI, 1989. Morgan Kaufmann.

[13] Daniel Dvorak and Benjamin Kuipers. Process Monitoring and Diagnosis: A Model-Based Approach. *IEEE Expert*, 5(3):67–74, June 1991.

[14] Daniel Luis Dvorak. *Monitoring and Diagnosis of Continuous Dynamic Systems using Semiquantitative Simulation.* PhD thesis, University of Texas at Austin, 1992.

[15] Adam Farquhar. *Automated Modeling of Physical Systems in the Presence of Incomplete Knowledge.* PhD thesis, University of Texas at Austin, 1993.

[16] Ira Joseph Haimowitz. *Knowledge-Based Trend Detection and Diagnosis.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1994.

[17] Rolf Isermann. Fault Diagnosis of Machines via Parameter Estimation and Knowledge Processing – Tutorial Paper. *Automatica*, 29(4):815–835, 1993.

[18] Herbert Kay. SQSIM: A Simulator for Imprecise ODE Models. *Computers and Chemical Engineering*, 23(1):27–46, 1998.

[19] Herbert Kay and Benjamin Kuipers. Numerical Behavior Envelopes for Qualitative Models. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pages 606–613, Cambridge, MA, 1993. AAAI/MIT Press.

[20] Herbert Kay and Lyle Ungar. Estimating Monotonic Functions and Their Bounds using MSQUID. Technical Report TR AI99-280, University of Texas at Austin, 1999.

[21] Herbert Kay and Lyle H. Ungar. Deriving Monotonic Function Envelopes from Observations. In *Working Papers from the Seventh International Workshop on Qualitative Reasoning about Physical Systems (QR-93)*, pages 117–123, Orcas Island, Washington, 1993.

[22] Benjamin Kuipers. Commonsense Reasoning about Causality: Deriving Behavior from Structure. In Daniel G. Bobrow, editor, *Qualitative Reasoning about Physical Systems*, pages 169–203. Elsevier Sience Publishers B.V., Amsterdam, Netherlands, 1984.

[23] Benjamin Kuipers. Qualitative Simulation. *Artificial Intelligence*, 29:289–338, 1986.

[24] Benjamin Kuipers. *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge.* Artificial Intelligence. MIT Press, 1994.

[25] Benjamin J. Kuipers and Daniel Berleant. Using incomplete knowledge with qualitative reasoning. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 324–329, Los Altos, CA, 1988. Morgan Kaufmann.

[26] Deepak Kulkarni, Kiriako Kutulakos, and Peter Robinson. Data analysis using scale-space filtering and bayesian probabilistic reasoning. Technical Report FIA-91-05, NASA Ames Research Center, 1991.

[27] Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall, Englewood Cliffs, New Jersey, 1987.

[28] Pieter J. Mosterman, Gautam Biswas, and Eric J. Manders. A Comprehensive Framework for Model Based Diagnosis. In *Proceedings of the Ninth International Workshop on Principles of Diagnosis (Dx98)*, pages 86–93, Cape Cod, MA, USA, May 1998.

[29] Hwee Tou Ng. Model-Based, Multiple-Fault Diagnosis of Dynamic, Continuous Physical Devices. *IEEE Expert*, 6(6):38–43, December 1991.

[30] Raymond Reiter. A Theory of Diagnosis from First Principles. *Artificial Intelligence*, 32:57–95, 1987.

[31] Bradley L. Richards, Ina Kraan, and Benjamin Kuipers. Automatic abduction of qualitative models. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, pages 723–728, 1992.

[32] Jeff Rickel and Bruce Porter. Automated Modeling for Answering Prediction Questions: Selecting the Time Scale and System Boundary. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 1191–1198, Seattle, Washington, 1994. AAAI.

[33] Bernhard Rinner and Benjamin Kuipers. Monitoring Piecewise Continuous Behaviors by Refining Semi-Quantitative Trackers. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 1080–1086, Stockholm, Sweden, August 1999. Morgan Kaufmann.

[34] A. C. Cem Say and Selahattin Kuru. Qualitative System Identification: Deriving Structure from Behavior. *Artificial Intelligence*, 83:75–141, 1996.

[35] Andrew P. Witkin. Scale-space Filtering. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 1019–1022, 1983.