Deriving Monotonic Function Envelopes from Observations*

Herbert Kay
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712
bert@cs.utexas.edu

Lyle H. Ungar
Department of Chemical Engineering
University of Pennsylvania
Philadelphia, PA 19104
ungar@cis.upenn.edu

Abstract

Much work in qualitative physics involves constructing models of physical systems using functional descriptions such as "flow monotonically increases with pressure." Semiquantitative methods improve model precision by adding numerical envelopes to these monotonic functions. Ad hoc methods are normally used to determine these envelopes. This paper describes a systematic method for computing a bounding envelope of a multivariate monotonic function given a stream of data. The derived envelope is computed by determining a simultaneous confidence band for a special neural network which is guaranteed to produce only monotonic functions. By composing these envelopes, more complex systems can be simulated using semiquantitative methods.

Introduction

Scientists and engineers build models of continuous systems to better understand and control them. Ideally, these models are constructed based on the the underlying physical properties of the system. Unfortunately, real systems are seldom well enough understood to construct precise models based solely on *a priori* knowledge of physical laws. Therefore, process data is often used to estimate some portions of the model.

Techniques for estimating a functional relationship between an output variable y and the input vector $\mathbf{x}$ typically assume that there is some deterministic function g and some random variate ϵ such that $y = g(\mathbf{x}) + \epsilon$, where ϵ is a normally distributed, mean zero random variable with variance σ^2 that represents measurement error and stochastic variations. The estimate is computed by using a parameterized fitting function $f(\mathbf{x};\boldsymbol{\theta})$ and then using regression analysis to determine the values $\widehat{\boldsymbol{\theta}}$ such that $f(\mathbf{x};\widehat{\boldsymbol{\theta}}) \approx g(\mathbf{x})$.

Traditional regression methods require knowledge of the form of the estimation function f. For instance, we may know that body weight is linearly related to amount of body fat. We may then determine that $f(weight;\theta) = weight \cdot \theta$ is an appropriate model. Neural network methods have been developed for cases where no information about f is known. This may be the case if f models a complex process whose physics is poorly understood. Such networks are known to be capable of representing any functional relationship given a large enough network. In this paper, we consider the case where some intermediate level of knowledge about f is known. In particular, we are interested in cases where we have knowledge of the monotonicity of f in terms of the signs of $\frac{\partial f}{\partial x_k}$ where $x_k \in \mathbf{x}$. For example, we might know that outflow from a tank monotonically increases with tank pressure. This type of knowledge is prevalent in qualitative descriptions of systems, so it makes sense to take advantage of it.

Since the estimate is based on a finite set of data, it is not possible for it to be exact. We therefore require our estimate to have an associated confidence measure which takes into account the uncertainty introduced by the finite sample size. For our semiquantitative representation, we are therefore interested in deriving an envelope that bounds all possible functions that could have generated the datastream with some probability p.

This paper describes a method for estimating and computing bounding envelopes for multivariate functions based on a set of data and knowledge of the monotonicity of the functional relationship. First, we describe our method for computing an estimate $f(\mathbf{x};\widehat{\boldsymbol{\theta}}) \approx g(\mathbf{x})$ based on a neural network that is constrained to produce only monotonic functions. Second, we describe our method for computing a bounding envelope, which is based on linearizing the estimation function and then using F-statistics to compute a simultaneous confidence band. Third, we present several examples of function fitting and its use in semiquantitative simulation using QSIM. Next we discuss related

*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-8905494, IRI-8904454, and IRI-9017047, by NASA contract NCC 2-760, and by the Jet Propulsion Laboratory.

work in neural networks and nonparametric analysis and finally we summarize the results and describe future work.

Bounded monotonic functions are a key element of the semiquantitative representation used by simulators such as Q2 [Kuipers and Berleant, 1988], Q3 [Berleant and Kuipers, 1992], and Nsim [Kay and Kuipers, 1993] which predict behaviors from models that are incompletely specified. To date, the bounds for such functions have been derived in an *ad hoc* manner. The work described in this paper provides a systematic method for finding these functional bounds. It is particularly appropriate for semiquantitative monitoring and diagnosis systems (such as MIMIC [Dvorak and Kuipers, 1989]) because process data is readily available in such applications. By combining our function bounding method with model abduction methods such as MISQ [Richards *et al.*, 1992], we plan to construct a self-calibrating system which derives models directly from observations of the monitored process. Such a system will have the property that as more data is acquired from the process, the model and its predictions will improve.

Computing the Estimate

Computing the estimate of g requires that we make some assumptions about the nature of deterministic and stochastic portions of the model. We assume that the relationship between y and $\mathbf{x}$ is $y = g(\mathbf{x}) + \epsilon$ where ϵ is a normally distributed random variable with mean 0 and variance σ^2. Other assumptions, such as different noise probability distributions or multiplicative rather than additive noise coupling could be made. The above model, however, is fairly general and it permits us to use powerful regression techniques for the computation of the estimate and its envelope. For situations where variance is not uniform, we can use variance stabilization techniques to transform the problem so that it has a constant variance.

In traditional regression analysis, the modeler supplies a function $f(\mathbf{x};\boldsymbol{\theta})$ together with a dataset to a least-squares algorithm which determines the optimal values for $\boldsymbol{\theta}$ so that $f(\mathbf{x};\widehat{\boldsymbol{\theta}}) \approx g(\mathbf{x})$. The estimated value of y is then $\hat{y} = f(\mathbf{x};\widehat{\boldsymbol{\theta}})$. In our case, however, the only information available about f is the signs of its n partial derivatives $\frac{\partial f}{\partial x_k}$ where $x_k \in \mathbf{x}$, so no explicit equation for f can be assumed. One way to work without an explicit form for f is to use a neural net function estimator. Figure 1 illustrates a network for determining $\hat{y}$ given a set of inputs $\mathbf{x}$. The network has three layers. The input layer contains one node for each element x_k and one bias node set to a constant value of 1. The hidden layer consists of a set of n_h nodes which are connected to each input variable as well as to the bias input. The output layer consists of a single node which is connected to all the hidden nodes as well as to another bias value which is fixed at 1[1]. All nodes use sigmoidal basis functions and all connections are weighted. In our notation, $w_{i[i,j]}$ represents the connection from input x_i to hidden node j and $w_{o[j,1]}$ represents the connection from hidden node j to the output layer[2]. This network represents the function

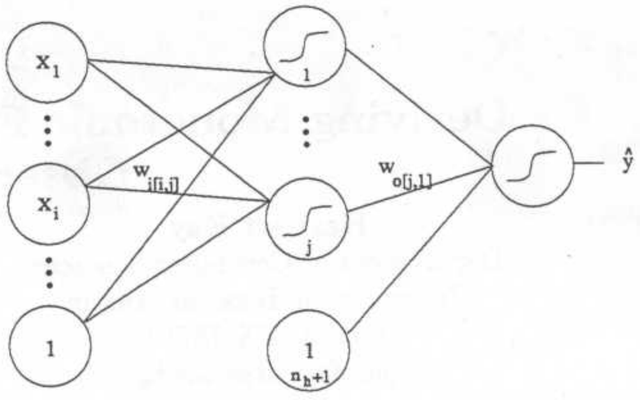

Figure 1: A neural net-based function estimator. This three-layer net computes the function $\hat{y} = \sigma\left(\sum_{j=1}^{n_h}\left[w_{o[j,1]}\sigma\left(\sum_{i=1}^{n} w_{i[i,j]}x_i + w_{i[n+1,j]}\right)\right] + w_{o[n_h+1,1]}\right)$.

$$
\begin{aligned}
\hat{y} &= \sigma(s + w_{o[n_h+1,1]}) \\
s &= \sum_{j=1}^{n_h}\left[w_{o[j,1]}\sigma\left(\sum_{i=1}^{n} w_{i[i,j]}x_i + w_{i[n+1,j]}\right)\right]
\end{aligned}
$$

where $\sigma(x)$ is the sigmoidal function $\frac{1-e^{-x}}{1+e^{-x}}$. We can compute the weights by solving the nonlinear least squares problem

$$
\min_{\mathbf{W}} \sum_i (y_i - \hat{y}_i)^2
$$

where $\hat{y}_i = f(\mathbf{x}_i;\mathbf{w})$ and $\mathbf{w}$ is a vector of all weights. Cybenko [Cybenko, 1989] and others have shown that with a large enough number of hidden units, any continuous function may be approximated by a network of this form.

One drawback of using this estimation function is that it can overfit the given data. This results in the estimate following the random variate ϵ as well as the deterministic part of the model which means that we get a poor approximation of g[3]. We therefore reduce the scope of possible functions to include only monotonic functions. To do this, note that if f is monotonically increasing in x_k, then $\frac{\partial f}{\partial x_k}$ must be positive.

[1] The bias terms permit the estimation function to shift the center of the sigmoid.

[2] The weight $w_{i[n+1,j]}$ represents the connection to the input bias and the weight $w_{o[n_h+1,1]}$ represents the connection from the hidden layer bias node to the output node.

[3] Visually, the estimate will try to pass through every datapoint.

By constraining the weights, we can force this derivative to positive for all $\mathbf{x}$, insuring that the resulting function is monotonic. The derivative in question is

$$\frac{\partial f}{\partial x_k} = \sigma'(s + w_{o[n_h+1,1]}) \cdot p$$

$$p = \sum_{j=1}^{n_h} w_{o[j,1]} \sigma' \left(\sum_{i=1}^{n} (w_{i[i,j]} x_i) + w_{i[k,j]} \right)$$

Since the derivative of the sigmoid function is positive for all values of its domain, $\frac{\partial f}{\partial x_k}$ will be positive if p is positive. This will be the case if $\forall_{1 \leq j \leq n} w_{o[j,1]} \cdot w_{i[k,j]} \geq 0$. If the partial derivative is negative, then the inequality is reversed.

This set of additional constraints on weights causes the network to produce only monotonic functions. We may determine the values of the weights by solving the problem

$$\min_{\mathbf{W}} \sum_i (y_i - \hat{y}_i)^2$$

$$\text{subject to } \forall_{\substack{1 \leq j \leq n_h \\ 1 \leq k \leq n}} w_{o[j,1]} \cdot w_{i[k,j]} \geq 0.$$

These $n_h \cdot n$ constraints transform the least squares problem into a constrained nonlinear optimization problem. While more difficult to solve than the unconstrained problem, there are still a number of solutions based on numerical methods. In our work we use the SQP algorithm [Biegler, 1985; Biegler and Cuthrell, 1985].

To compute the estimate, we need to find the best value for n_h. We determine n_h by repeatedly solving the optimization problem for increasing values of n_h starting at 1 and continuing until there is no improvement in the sample standard error

$$\frac{\sum_i (y_i - \hat{y}_i)^2}{n - p}.$$

For our examples, n_h runs around 2.

Figure 2 shows the result of using the above method to estimate a fit to a datastream derived from a quadratic function with noise from a normal distribution with $\sigma^2 = 4$. Note that the estimate is in fact monotonic and does not follow the noise.

Computing the Envelope

The estimate computed in the previous section is affected by the sample dataset. Since this dataset is of finite size, the estimate generated from it will not be precisely correct. In this section, we describe our method for bounding our estimate with an envelope that captures the uncertainty introduced by using a finite set of data.

The typical confidence estimate used in regression analysis is the confidence interval, which is defined at a point $\mathbf{x}$ as $P(f(\mathbf{x}) - b_{\mathbf{x}} \leq g(\mathbf{x}) \leq f(\mathbf{x}) + b_{\mathbf{x}}) = 1 - \alpha$

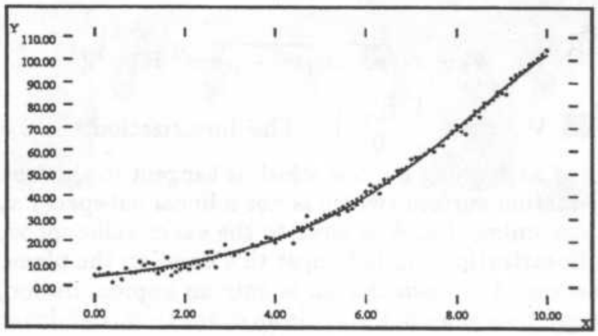

Figure 2: Fitting a neural net estimator to a datastream of 100 points. The data was generated by adding noise with a variance of 4 to the line $y = x^2 + 5$ (shown as a dashed line). The estimated function is shown as a solid line. There are two hidden nodes.

where $b_{\mathbf{x}}$ depends on $\mathbf{x}$. The confidence interval is a point probability measure since it expresses the uncertainty of the estimated value $f(\mathbf{x})$ at a single point $\mathbf{x}$ in the domain. Since we wish our envelope to bound all possible functions, we require that our confidence interval holds simultaneously at all points in the domain. This measure (called a *confidence band*) can be easily computed for linear regression models. Since our model is nonlinear, we use a linearization of f to form an approximate confidence band.

Assume that we have a linear model $f(\mathbf{x}; \boldsymbol{\beta}) = \mathbf{x}^{\mathbf{T}} \boldsymbol{\beta}$ where $\boldsymbol{\beta}$ is a parameter vector of length p and that $\widehat{\boldsymbol{\beta}}$ is our estimate of the parameters. If we represent the datastream as a matrix $[\mathbf{Y} \mid \mathbf{X}]$ where the ith row represents a single sample datapoint $(y_i, \mathbf{x}_i)$, it can be shown [Bates and Watts, 1988] that the $1 - \alpha$ confidence band for f is

$$\mathbf{x}^{\mathbf{T}} \widehat{\boldsymbol{\beta}} \pm s\sqrt{pF(\alpha; p, n-p)} \|\mathbf{x}^{\mathbf{T}} \mathbf{R}^{-1}\|$$

where s^2 is the sample standard error $\frac{\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\|^2}{n-p}$, F is the α quantile of the F-statistic with p and $n - p$ degrees of freedom, and $\mathbf{R}$ is the square portion of the QR decomposition of the array of sample inputs $\mathbf{X}$ ($\mathbf{X} = \mathbf{Q} \begin{bmatrix} \mathbf{R} \\ 0 \end{bmatrix}$). Geometrically, the columns of $\mathbf{X}$ form a p-dimensional linear subspace called the *expectation surface* in which the solution must lie. The least squares computation finds the point on the expectation surface that is closest to $\mathbf{Y}$.

To use this result, we must linearize the nonlinear problem as follows. Note that the entry x_{np} of $\mathbf{X}$ is $\frac{\partial \mathbf{x}_n^{\mathbf{T}}}{\partial \beta_p}$. Using this, we linearize our estimator by defining a matrix $\mathbf{V}$ such that $v_{np} = \frac{\partial f(\mathbf{x}_n; \hat{\mathbf{W}})}{\partial w_p}$ and a vector $\mathbf{v}$ such that $v_p = \frac{\partial f(\mathbf{x}; \hat{\mathbf{W}})}{\partial w_p}$. The envelope is then defined by

$$f(\mathbf{x};\hat{\mathbf{w}}) \pm s\sqrt{pF(\alpha;p,n-p)}\|\mathbf{v}^{\mathbf{T}}\mathbf{R_V}^{-1}\|$$

where $\mathbf{V} = \mathbf{Q_V}\begin{bmatrix}\mathbf{R_V}\\ 0\end{bmatrix}$. The linearization can be viewed as defining a plane which is tangent to the true expectation surface (which is not a linear subspace) at $\hat{\mathbf{w}}$. Assuming that $\hat{\mathbf{w}}$ is close to the exact value for $\mathbf{w}$, the linearization will hold near this point on the plane.

Because the linearization is only an approximation, this estimate does not provide an exact $1-\alpha$ confidence band for f. However, the result is approximately correct and depends on the degree of nonlinearity of the expectation surface [Bates and Watts, 1988].

We use the LINPACK subroutine DQRDC to compute the QR decomposition of $\mathbf{V}$. This method performs the decomposition using pivoting so that $\mathbf{R_V}$ is both upper triangular and has diagonal elements whose magnitude decreases going down the diagonal. With this form, we can easily recognize the case where the linearized matrix $\mathbf{V}$ may not have full rank (i.e., only $q < p$ columns are linearly independent). This in turn means that $\mathbf{R_V}$ has zeros in all diagonals past column q. In such cases, the product $\mathbf{v}^{\mathbf{T}}\mathbf{R_V}^{-1}$ may not have a solution. To handle this problem, we simply reduce the size of $\mathbf{R_V}$ by using only the upper left $q \times q$ corner. We must then also reduce the size of $\mathbf{v}$ so that it does not contain partials with respect to w_i where $i > q$. This is justified, since if $\mathbf{V}$ forms a p-dimensional basis for the linearized expectation surface, and its rank is only q, then the w_i where $i > q$ may be ignored since their value is arbitrary.

The confidence band is designed to cover all possible curves that fit the data with a probability of $1-\alpha$. With traditional regression, this confidence band is the most precise description we can achieve given the general form of f. Using monotonicity information, however, we can further refine our prediction to derive a tighter envelope by ruling out portions of the curves that could not be monotonic. Consider the confidence band shown in Figure 3. If we know that the underlying function is monotonic we may remove the shaded regions from the prediction since no monotonic function within the confidence band could pass through these regions. Note that this final step could not be performed if we used point confidence intervals.

Examples

We have applied our envelope method to noisy datasets whose underlying functions are linear, quadratic, square root, and exponential. In order to give a feeling for the form of the envelopes generated, we present several of these test cases. Each result is for a univariate monotonically increasing model function $f(x)$. Figure 4 shows the estimate and envelope for a set of 100 samples drawn from the function $y = 0.5x + 5$ with additive noise with variance 4. Note that the envelope

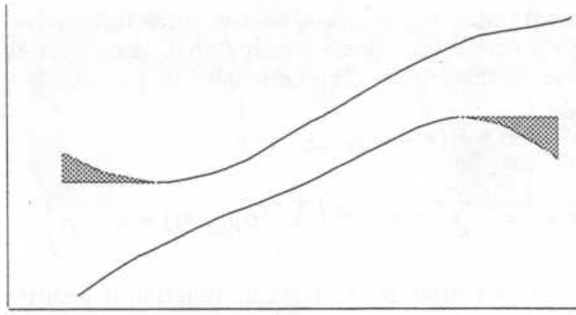

Figure 3: Reducing the size of a confidence band using monotonicity information. If we know that the underlying function is monotonically increasing, we can rule out the shaded areas of the figure as part of the envelope.

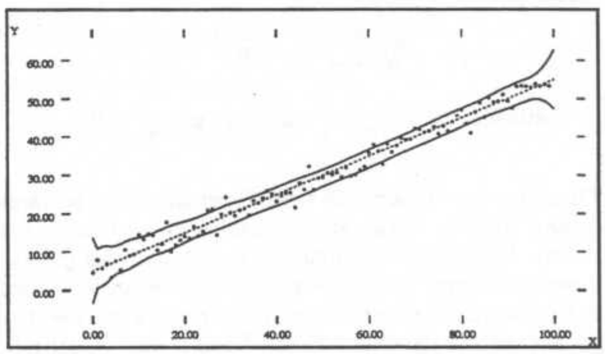

Figure 4: 95% envelope (solid lines) for the linear function $y = 0.5x + 5$ (dashed line). The estimator has 3 hidden nodes. Sample variance is 4.520.

expands at the ends since there is less data there to constrain the estimate.

The second example is that of a quadratic function with noise of variance 4. This dataset demonstrates the effect of nonuniform sampling. In areas where there is little data, the envelope bulges outward to compensate. Note also that the upper portion of the envelope is narrowest at the far end of the plot. This is due to bias in the estimation function which assumes that the curve will "heel over" just past the end of the dataset. For this reason, the computed envelope is valid only over the range of the given data.

In the third example, the envelope method is used to create a semiquantitative model which is then simulated using QSIM to predict the behavior of a real physical system. To develop this example, we took a small plastic tank and drilled a hole in its bottom. Our goal was to construct a semiquantitative model that could be used to predict the level of fluid in the tank over time. The standard approach to constructing such a model in QSIM is to construct a qualitative

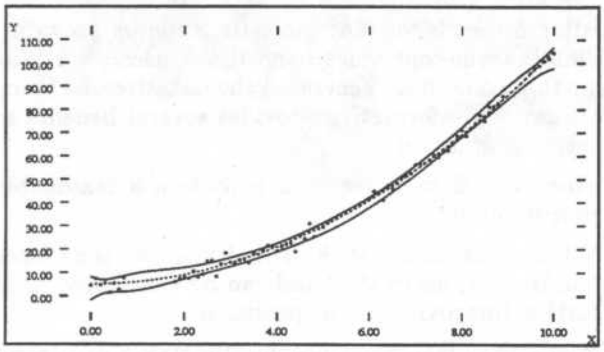

Figure 5: 95% envelope (solid lines) for the quadratic function $y = x^2 + 5$ (dashed line). The estimator has two hidden nodes. Sample variance is 4.227.

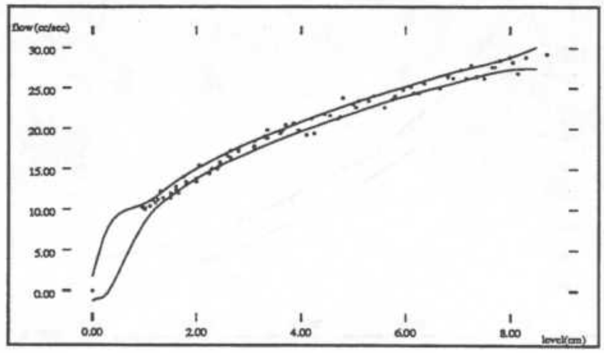

Figure 6: 95% envelope for level vs flow data from a tank. The estimator has two hidden nodes. Sample variance is 0.441.

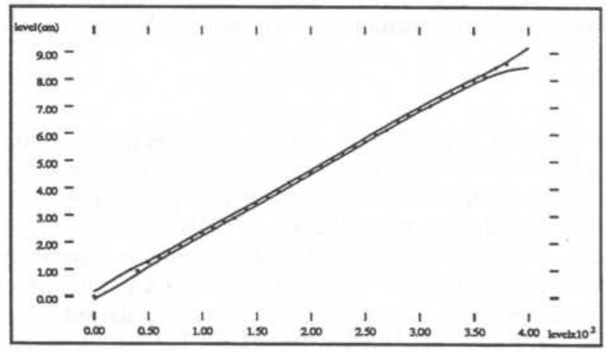

Figure 7: 95% envelope for the amount vs level data. The measurement accuracy for this data was greater than for the level vs flow data, hence the estimate is narrower. The estimator has two hidden nodes. Sample variance is 0.003.

model and then augment it with numerical information about the monotonic functions and parameters. Normally, these numerical envelopes are *ad hoc* in that they are hand-derived by the modeler. By applying the envelope method to a data stream from the system, we can construct these envelopes in a more principled way.

We require that the model prediction bound all real behaviors of the system (i.e., that the true behavior is within the predicted bounds) with probability $1 - \alpha$. Semiquantitative models are particularly good at this because they produce a bounded solution. We begin by asserting the qualitative part of the model – $A' = -f(A)$, $f \in M^+$ which asserts that the derivative of amount (A) is a monotonic function of amount. Since we are interested in level rather than amount, we also assert that $g(L) = A$, $g \in M^+$, i.e., that level (L) and amount are monotonically related. The complete qualitative model is thus

$$A' = -f(g(L)) \qquad f, g \in M^+$$

This qualitative description holds for a wide class of tanks. While it is unquestionably true for our tank, it is not specific enough to produce useful predictions. To increase the precision of the prediction, we must further define $f(A)$ and $g(L)$. To determine these functions, we performed the following experiment. We filled the tank with water and allowed it to drain while measuring flow (cc/sec), level (cm), and amount (cc). Figures 6 and 7 show the experimental data together with the 95% confidence envelopes. With this information, we used the QSIM semiquantitative simulation techniques Q2 [Kuipers and Berleant, 1988] and Nsim [Kay and Kuipers, 1993] to produce a prediction. Since QSIM already reasons with monotonic envelopes, there are no special techniques required to handle the composed function $f(g(L))$.

Figure 8 shows the results of the simulation together with experimental data for time vs level. Since the width of the monotonic function envelopes are a function of the datastream length, by observing the physical system for a longer time period, we would expect the envelopes to tighten further, thus yielding tighter predictions.

Related Work

This work is related to several approaches to function estimation using neural networks. Cybenko [Cybenko, 1989] has shown that any continuous function can be represented with a neural net that is similar to one that we use. With our method, we assume a priori that the true function is monotonic. This assumption restricts our attention to a smaller class of functions which means that less data is needed to compute a reasonable estimate. Additionally, we compute a confidence measure on our estimate in the form of an envelope.

The VI-NET method [Leonard *et al.*, 1992] also uses a neural network for computing estimates of general

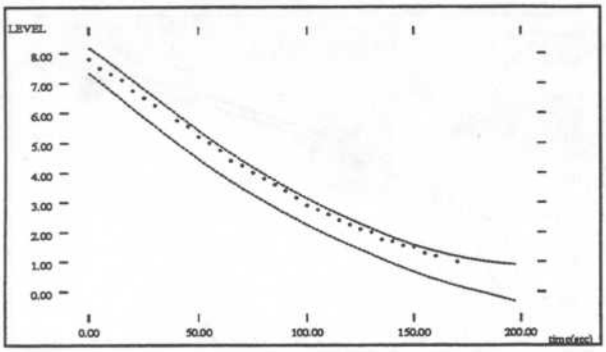

Figure 8: Prediction envelope for the tank model using the previous monotonic envelopes for f and g. Note that sample measurements taken from the system are all contained within the envelope. The predictions of this model could be strengthened with increased measurements which would serve to reduce the monotonic envelopes.

functions. It is notable in that it provides a confidence measure on its estimate and can determine when it is being asked to extrapolate. By using radial basis functions instead of sigmoidal ones, it is able to handle different variances across the function. This is especially useful in applications where there is no a priori information about g. Because it allows non-constant variances, it would be difficult to get a VI-NET to return simultaneous confidence bands, which are important since we wish to bound all curves that could have generated the datastream. In contrast, our approach can only handle fixed-variance problems, but since variance will often track either $\mathbf{x}$ or y, we can make monotonicity assumptions about σ^2 if it should prove necessary. Under such circumstances, variance stabilization techniques [Draper and Smith, 1981] should prove useful in fixing the variance.

This work is also related to monotonic function estimation [Kruskal, 1964; Hellerstein, 1990], particularly the NIMF estimator [Hellerstein, 1990] which also determines envelopes for multivariate monotonic functions. NIMF allows for a more general noise model (zero mean, *symmetrically* distributed) and uses a non-parametric statistical method for determining point confidence bounds. These bounds, however, are much weaker than those derived with methods that first compute an estimate. This is not surprising, since we are assuming more about the noise than NIMF does. Finally, NIMF produces point bounds only, and it is unclear how these bounds could be made into confidence bands of reasonable width.

Discussion and Future Work

This paper has described a method for computing envelopes for functions described solely by monotonicity information and a stream of data. It improves over existing methods for function estimation by providing a simultaneous confidence band that encloses all functions that could have generated the datastream. Using monotonicity information provides several benefits to function estimation :

- Much less data is required to obtain a reasonably precise model.
- When combined with simultaneous confidence bands, portions of the band can be eliminated, thus further improving model precision.
- While the class of monotonic functions may seem restrictive for modeling continuous systems, we can represent many non-monotonic functions as compositions of monotonic ones. For example, in a system of two cascaded tanks, the level of water in the lower tank is typically a non-monotonic function of time. The flow into the lower tank, however, is the composition of a monotonically increasing function of the amount of water in the upper tank and a monotonically decreasing function of the water in the lower tank. By using semiquantitative simulation methods, we can represent this composition and thus simulate systems with non-monotonic behaviors.

The method is applicable for cases where sample variance is fixed. Future work includes adding variance stabilization methods to handle cases where the data has a non-constant variance. We also plan to add the ability to impose further constraints such as stating that the function must pass through zero.

Our method for deriving bounds for monotonic functions plays a key role in the construction of semiquantitative models, especially for monitoring and diagnosis tasks where process data is readily available. Because the precision of the resulting envelopes is a function of the amount of data used to compute them, our bounding method also provides a systematic method for shifting the precision of a semiquantitative model along a continuum from purely qualitative to exact.

Nomenclature

$\mathbf{x}$	The domain of the function ($y = g(\mathbf{x})$).
n	The dimension of $\mathbf{x}$.
n_h	The number of hidden units in the network.
p	The number of weights in the network (a total of $(n+2)n_h + 1$).
$w_{i[i,j]}$	The weight from x_i to hidden unit j.
$w_{o[j,1]}$	The weight from hidden unit j to the output.
$\mathbf{w}$	A vector of all weights.
$\hat{y}$	The estimate of y.

References

Bates, Douglas M. and Watts, Donald G. 1988. *Nonlinear Regression and Its Applications.* John Wiley and Sons, New York.

Berleant, Daniel and Kuipers, Benjamin 1992. Combined qualitative and numerical simulation with q3.

In Faltings, Boi and Struss, Peter, editors 1992, *Recent Advances in Qualitative Physics.* MIT Press.

Biegler, L. T. and Cuthrell, J. E. 1985. Improved infeasible path optimization for sequential modular simulators – II. the optimization algorithm. *Computers and Chemical Engineering* 9(3):257–267.

Biegler, L. T. 1985. Improved infeasible path optimization for sequential modular simulators – I. the interface. *Computers and Chemical Engineering* 9(3):245–256.

Cybenko, G. 1989. Approximation by superpositions of sigmoidal functions. *Mathematics of Control, Signals, and Systems* 2:303–314.

Draper, Norman R. and Smith, H. 1981. *Applied Regression Analysis (second edition).* John Wiley and Sons, New York.

Dvorak, Daniel Louis and Kuipers, Benjamin 1989. Model-based monitoring of dynamic systems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence.* 1238–1243.

Hellerstein, Joseph 1990. Obtaining quantitative predictions from monotone relationships. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90).* 388–394.

Hertz, John; Krogh, Anders; and Palmer, Richard G. 1991. *Introduction to the Theory of Neural Computation.* Addison-Wesley, Redwood City.

Kay, Herbert and Kuipers, Benjamin 1993. Numerical behavior envelopes for qualitative models. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93).* to appear.

Kruskal, J. B. 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika* 29(1):1–27.

Kuipers, Benjamin and Berleant, Daniel 1988. Using incomplete quantitative knowledge in qualitative reasoning. In *Proceedings of the Seventh National Conference on Artificial Intelligence.* 324–329.

Leonard, J. A.; Kramer, M. A.; and Ungar, L. H. 1992. Using radial basis functions to approximate a function and its error bounds. *IEEE Transactions of Neural Networks* 3(4):624–627.

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