Guaranteed State and Parameter Estimation for Nonlinear Continuous-Time Systems with Bounded-Error Measurements

Youdong Lin and Mark A. Stadtherr*

Department of Chemical and Biomolecular Engineering
University of Notre Dame, Notre Dame, IN 46556, USA

June 4, 2007
(revised, July 24, 2007)

*Author to whom all correspondence should be addressed. Phone: (574) 631-9318; Fax: (574) 631-8366; E-mail: markst@nd.edu

Abstract

A strategy for state and parameter estimation in nonlinear, continuous-time systems is presented. The method provides guaranteed enclosures of all state and parameter values that are consistent with bounded-error output measurements. Key features of the method are the use of a new validated solver for parametric ODEs, which is used to produce guaranteed bounds on the solutions of nonlinear dynamic systems with interval-valued parameters and initial states, and the use of a constraint propagation strategy on the Taylor models used to represent the solutions of the dynamic system. Numerical experiments demonstrate the use and computational efficiency of the method.

Keywords: State estimation; Parameter estimation; Continuous-time systems; Validated methods; Interval methods

1 Introduction

In this paper we consider the estimation of state variables and parameters for nonlinear, continuous-time systems in a bounded-error context. This problem was first addressed by Jaulin,[1] though other versions of the problem (e.g., with linear models and/or discrete time) have also been studied.[2–5] Jaulin[1] proposed an algorithm for state estimation based on interval analysis[6] and consistency (constraint propagation) techniques.[7] A first-order method was used to get a guaranteed enclosure of the solution of the ordinary differential equations (ODEs) describing the system. Consistency techniques then were used to contract the domains for the state variables by pruning parts that are inconsistent with the measured output. However, the large wrapping effect associated with the first-order method leads to very pessimistic results. Raïssi et al.[8] provided a technical improvement by using a more accurate interval computation of the solution of the ODEs. Use of a high-order Taylor series method, together with other techniques proposed by Rihm,[9] made it possible to obtain a better enclosure of the solution to the ODE system. This approach was also used to provide a method for guaranteed parameter estimation.

These approaches rely on the use of interval methods[6] (also called validated or verified methods) for solving the initial value problem (IVP) for ODEs. When the parameters and/or initial states are not known with certainty and are given by intervals, traditional approximate solution methods for ODEs are not useful, since, in essence, they would have to solve infinitely many systems to determine an enclosure of the solutions. In contrast, interval methods not only can determine a guaranteed error bound on the true solution, but can also verify that a unique solution to the problem exists. An excellent review of interval methods for IVPs has been given by Nedialkov et al.[10] For addressing this problem, there are various packages available, including AWA,[11] VNODE,[12] and COSY VI,[13] all of which consider uncertainties in initial state only.

In this study, we use a new package named VSPODE (Validating Solver for Parametric ODEs), described by Lin and Stadtherr,[14] for computing a validated enclosure of all solutions of an ODE system with interval-valued parameters and/or initial state. The method is based on a traditional interval approach,[10] but involves a novel use of Taylor models[15,16] to address the dependency

problem in interval arithmetic. In addition, a constraint propagation[2,19] procedure based on Taylor models is employed to efficiently reduce the domain of the uncertain quantities, which provides significant advantages over the methods used by Jaulin[1] and Raïssi et al.[8] The use of Taylor models for state and parameter estimation in this context was first suggested by Lin and Stadtherr,[17] and again more recently by Kletting et al.[18]

The rest of this paper is organized as follows. A problem statement is given in Section 2. Background on interval analysis and Taylor models is presented in Section 3. The validating solver for parametric ODEs is described in Section 4. The guaranteed nonlinear state and parameter estimator is then presented in Section 5. Finally in Section 6, we present the results of some numerical experiments that demonstrate the effectiveness of the new estimation method.

2 Problem Statement

Consider a nonlinear, continuous-time system represented by the following equations:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \tag{1}$$

$$\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\theta}), \tag{2}$$

where $\boldsymbol{x}$ is the m-dimensional state vector, $\boldsymbol{\theta}$ is a p-dimensional time-invariant parameter vector, and $\boldsymbol{y}$ is the n-dimensional output vector. Output measurements $\widehat{\boldsymbol{y}}_j$ at $t = t_j$ are available with error $\boldsymbol{v}_j = \widehat{\boldsymbol{y}}_j - \boldsymbol{y}_j$, where $\boldsymbol{y}_j = \boldsymbol{g}(\boldsymbol{x}_j, \boldsymbol{\theta})$ and $\boldsymbol{x}_j = \boldsymbol{x}(t_j)$. A sequence of N sampling times $t_0 < t_1 < \cdots < t_N$, at which the measurements $\widehat{\boldsymbol{y}}_j$ have been collected, is considered. We assume that $\boldsymbol{f}$ is $(k-1)$-times continuously differentiable with respect to the state variables $\boldsymbol{x}$. Here k is the order of the truncation error in the interval Taylor series (ITS) method to be used in the integration procedure (to be discussed in Section 4). We also assume that $\boldsymbol{f}$ and $\boldsymbol{g}$ are $(q+1)$-times continuously differentiable with respect to the uncertain quantities (initial states $\boldsymbol{x}_0$ and/or parameters $\boldsymbol{\theta}$), where q is the order of the Taylor model to be used to represent initial state and parameter dependence (to be discussed in Section 3.2).

In this study, the initial state $\boldsymbol{x}_0$ is assumed to belong to a known interval $\boldsymbol{X}_0$. The parameter

vector $\boldsymbol{\theta}$ is assumed to be constant and belong to a known interval $\boldsymbol{\Theta}$. Both $\boldsymbol{X}_0$ and $\boldsymbol{\Theta}$ could represent arbitrarily large domains. The measurement error $\boldsymbol{v}_j$ is bounded and assumed to belong to a known interval $\boldsymbol{V}_j$ at each t_j. Therefore, the output vector $\boldsymbol{y}_j$ belongs to a known interval $\boldsymbol{Y}_j = \widehat{\boldsymbol{y}}_j - \boldsymbol{V}_j$. The goal is then to use this information to obtain estimates of the state vector $\boldsymbol{x}_j$ at each measurement time $j = 0, \ldots, N$ and of the parameter vector $\boldsymbol{\theta}$.

3 Background

3.1 Interval Analysis

A real interval X is defined as the set of real numbers lying between (and including) given upper and lower bounds; that is,

$$X = \left[\underline{X}, \overline{X}\right] = \left\{x \in \mathbb{R} \mid \underline{X} \leq x \leq \overline{X}\right\}. \tag{3}$$

Here an underline is used to indicate the lower bound of an interval and an overline is used to indicate the upper bound. A real interval vector $\boldsymbol{X} = (X_1, X_2, \cdots, X_n)^T$ has n real interval components and can be interpreted geometrically as an n-dimensional rectangle or box. Unless indicated otherwise, uppercase quantities are intervals, and lowercase quantities or uppercase quantities with underline or overline are real numbers.

Basic arithmetic operations with intervals are defined by

$$X \text{ op } Y = \{x \text{ op } y \mid x \in X, y \in Y\}, \tag{4}$$

where op $= \{+, -, \times, \div\}$. That is, the result of an interval arithmetic operation on X and Y is an interval enclosing the range of results obtainable by performing the operation with any number in X and any number in Y. Interval versions of the elementary functions can be similarly defined. For dealing with exceptions, such as division by an interval containing zero, extended models for interval arithmetic are available, often based on the extended real system $\mathbb{R}^* = \mathbb{R} \cup \{-\infty, +\infty\}$. The concept of containment sets (csets) provides a valuable framework for constructing models for interval arithmetic with consistent handling of exceptions.[19,20] When machine computations

using intervals are performed, rounding errors must be handled correctly in order to ensure that the result is a rigorous enclosure. Thus, when machine computations with interval arithmetic operations are done, as in the procedures outlined below, the endpoints of an interval are computed with a directed (outward) rounding. That is, the lower endpoint is rounded down to the next machine-representable number and the upper endpoint is rounded up to the next machine-representable number. Several good introductions to interval analysis, as well as interval arithmetic and other aspects of computing with intervals, are available.[2,19,21,22] Implementations of interval arithmetic and elementary functions are also readily available, and recent compilers from Sun Microsystems directly support interval arithmetic and an interval data type.

For an arbitrary function $f(\boldsymbol{x})$, the interval extension, $F(\boldsymbol{X})$ encloses all possible values of $f(\boldsymbol{x})$ for $\boldsymbol{x} \in \boldsymbol{X}$; that is, it encloses the range of $f(\boldsymbol{x})$ over $\boldsymbol{X}$. It is often computed by substituting the given interval $\boldsymbol{X}$ into the function $f(\boldsymbol{x})$ and then evaluating the function using interval arithmetic. This "natural" interval extension is often wider than the actual range of function values, though it always includes the actual range. This potential overestimation of the function range is due to the "dependency" problem, which may arise when a variable occurs more than once in a function expression. While a variable may take on any value within its interval, it must take on the *same* value each time it occurs in an expression. However, this type of dependency is not recognized when the natural interval extension is computed. In effect, when the natural interval extension is used, the range computed for the function is the range that would occur if each instance of a particular variable were allowed to take on a different value in its interval range. For the case in which $f(\boldsymbol{x})$ is a single-use expression, that is, an expression in which each variable occurs only once, natural interval arithmetic will always yield the true function range. For cases in which obtaining a single-use expression is not possible, there are several other approaches that can be used to tighten interval extensions,[2,19,21–23] including the use of Taylor models, as described in the next subsection.

In some situations, including one encountered in Section 5.1, dependency issues can be avoided through the use of the dependent subtraction operation (also known as the cancellation operation). Assume that there is an interval S that depends additively on the interval A. The dependent subtraction operation is defined by $S \ominus A = [\underline{S} - \underline{A}, \overline{S} - \overline{A}]$. For example, let $A = [1, 2]$, $B = [2, 3]$,

$C = [3,4]$ and $S = A + B + C = [6,9]$. Say that only S is stored and that later it is desired to compute $A + B$ by subtracting C from S. Using the standard subtraction operation yields $S - C = [6,9] - [3,4] = [2,6]$, which overestimates the true $A + B$. Using the dependent subtraction operation, which is allowable since S depends additively on C, yields $S \ominus C = [6,9] \ominus [3,4] = [3,5]$, which is the true $A + B$.

3.2 Taylor Models

Makino and Berz have described a remainder differential algebra (RDA) approach that uses Taylor models for bounding function ranges and control of the dependency problem of interval arithmetic.[15,16] In this method, a function is represented using a model consisting of a Taylor polynomial and an interval remainder bound.

One way of forming a Taylor model of a function is by using a truncated Taylor series. Consider a function $f : \boldsymbol{x} \in \boldsymbol{X} \subset \mathbb{R}^m \to \mathbb{R}$ that is $(q+1)$ times partially differentiable on $\boldsymbol{X}$ and let $\boldsymbol{x}_0 \in \boldsymbol{X}$. The Taylor theorem states that for each $\boldsymbol{x} \in \boldsymbol{X}$, there exists a $\zeta \in \mathbb{R}$ with $0 < \zeta < 1$ such that

$$\begin{aligned} f(\boldsymbol{x}) \;=\; & \sum_{i=0}^{q} \frac{1}{i!} \left[(\boldsymbol{x} - \boldsymbol{x}_0) \cdot \bigtriangledown\right]^i f\left(\boldsymbol{x}_0\right) \\ + \; & \frac{1}{(q+1)!} \left[(\boldsymbol{x} - \boldsymbol{x}_0) \cdot \bigtriangledown\right]^{q+1} f\left[\boldsymbol{x}_0 + (\boldsymbol{x} - \boldsymbol{x}_0)\zeta\right], \end{aligned} \tag{5}$$

where the partial differential operator $[\boldsymbol{g} \cdot \bigtriangledown]^k$ is

$$[\boldsymbol{g} \cdot \bigtriangledown]^k = \sum_{\substack{j_1 + \cdots + j_m = k \\ 0 \le j_1, \cdots, j_m \le k}} \frac{k!}{j_1! \cdots j_m!} g_1^{j_1} \cdots g_m^{j_m} \frac{\partial^k}{\partial x_1^{j_1} \cdots \partial x_m^{j_m}}. \tag{6}$$

The last (remainder) term in eq 5 can be quantitatively bounded over $0 < \zeta < 1$ and $\boldsymbol{x} \in \boldsymbol{X}$ using interval arithmetic or other methods to obtain an interval remainder bound R_f. The summation in eq 5 is a q-th order polynomial (truncated Taylor series) in $(\boldsymbol{x} - \boldsymbol{x}_0)$ which we denote by $p_f(\boldsymbol{x} - \boldsymbol{x}_0)$. A q-th order Taylor model T_f for $f(\boldsymbol{x})$ then consists of the polynomial p_f and the interval remainder bound R_f and is denoted by $T_f = (p_f, R_f)$. Note that $f \in T_f$ for $\boldsymbol{x} \in \boldsymbol{X}$ and thus T_f encloses the range of f over $\boldsymbol{X}$.

In practice, it is more useful to compute Taylor models of functions by performing Taylor model operations. Arithmetic operations with Taylor models can be done using the RDA operations

described by Makino and Berz,[15,16,24,25] which include addition, multiplication, reciprocal, and intrinsic functions. Using these, it is possible to start with simple functions such as the constant function $f(x) = k$, for which $T_f = (k, [0, 0])$, and the identity function $f(x_i) = x_i$, for which $T_f = (x_{i0} + (x_i - x_{i0}), [0, 0])$, and then to compute Taylor models for very complicated functions. Therefore, it is possible to compute a Taylor model for any function representable in a computer environment by simple operator overloading through RDA operations.

In performing RDA operations, only the coefficients of the polynomial part p_f are used, and these are point valued. However, when these coefficients are computed in floating point arithmetic, numerical errors may occur and they must be bounded. To do this in our current implementation of Taylor model arithmetic, we have used the "tallying variable" approach, as described by Makino and Berz.[25] This approach has been analyzed in detail by Revol et al.[26] This results in an error bound on the floating point calculation of the coefficients in p_f being added to the interval remainder bound R_f. It has been shown that, compared to other rigorous bounding methods, the Taylor model often yields sharper bounds for modest to complicated functional dependencies.[15,16,27] A discussion of the uses and limitations of Taylor models has been given by Neumaier.[27]

An interval bound on a Taylor model $T = (p, R)$ over $\boldsymbol{X}$ is denoted by $B(T)$, and is found by determining an interval bound $B(p)$ on the polynomial part p and then adding the remainder bound; that is $B(T) = B(p) + R$. The range bounding of the polynomial $B(p) = P(\boldsymbol{X} - \boldsymbol{x}_0)$ is an important issue, which directly affects the performance of Taylor model methods. Unfortunately, exact range bounding of an interval polynomial is NP hard, and direct evaluation using interval arithmetic is very inefficient, often yielding only loose bounds. Thus, various bounding schemes[27–29] have been used, mostly focused on exact bounding of the dominant parts of P, i.e., the first- and second-order terms. However, exact bounding of a general interval quadratic is also computationally expensive (in the worst case, exponential in the number of variables m). Thus, we have adopted a very simple compromise approach, in which only the first-order and the diagonal second-order terms are considered for exact bounding, and other terms are evaluated directly. That is,

$$B(p) = \sum_{i=1}^{m} \left[a_i \left(X_i - x_{i0} \right)^2 + b_i (X_i - x_{i0}) \right] + Q, \tag{7}$$

where Q is the interval bound of all other terms, and is obtained by direct evaluation with interval arithmetic. In eq 7, since X_i occurs twice, there exists a dependency problem. However, we can rearrange eq 7 such that each X_i occurs only once; that is,

$$B(p) = \sum_{i=1}^{m} \left[a_i \left(X_i - x_{i0} + \frac{b_i}{2a_i} \right)^2 - \frac{b_i^2}{4a_i} \right] + Q. \tag{8}$$

In this way, the dependency problem in bounding the interval polynomial is alleviated so that a sharper bound can be obtained. Since we prefer not to divide by a very small number, eq 8 will be used only if $|a_i| \geq \omega$, where ω is a very small positive number. If $|a_i| < \omega$, direct evaluation with eq 7 will be used instead.

4 Validating Solver for Parametric ODEs

Interval methods[6] (also called validated or verified methods) for ODEs not only can determine guaranteed bounds on the state variables, but can also verify that a unique solution to the problem exists. Traditional interval methods usually consist of two processes applied at each integration step.[6] In the first process, existence and uniqueness of the solution are proved using the Picard-Lindelöf operator and the Banach fixed point theorem,[30] and a rough enclosure of the solution is computed. In the second process, a tighter enclosure of the solution is computed. In general, both processes are realized by applying interval Taylor series (ITS) expansions with respect to time, and using automatic differentiation to obtain the Taylor coefficients. An excellent review of the traditional interval methods has been given by Nedialkov et al.[10] For addressing this problem, there are various packages available, including AWA,[11] VNODE[12] and COSY VI,[13] all of which consider uncertainties (interval valued) in initial values only. In this study, we will use a new validated solver[14] for parametric ODEs, which is used to produce guaranteed bounds on the solutions of nonlinear dynamic systems with interval-valued initial states and parameters. The method makes use, in a novel way, of the Taylor model approach[15,16,25] to deal with the dependency problem on the uncertain variables (parameters and initial values). We will summarize here the basic ideas of the method used. Additional background and details are given by Lin and Stadtherr.[14]

Consider the following parametric ODE system, with state variables and parameters denoted by $\boldsymbol{x}$ and $\boldsymbol{\theta}$, respectively:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}), \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \in \boldsymbol{X}_0, \quad \boldsymbol{\theta} \in \boldsymbol{\Theta}, \tag{9}$$

where $t \in [t_0, t_N]$ for some $t_N > t_0$. The interval vectors $\boldsymbol{X}_0$ and $\boldsymbol{\Theta}$ represent enclosures of initial values and parameters, respectively. It is desired to determine a validated enclosure of all possible solutions to this initial value problem. Also note that nonautonomous (time dependent) problems can be converted to the autonomous form given in eq 9. We denote by $\boldsymbol{x}(t; t_j, \boldsymbol{X}_j, \boldsymbol{\Theta})$ the set of solutions $\boldsymbol{x}(t; t_j, \boldsymbol{X}_j, \boldsymbol{\Theta}) = \{\boldsymbol{x}(t; t_j, \boldsymbol{x}_j, \boldsymbol{\theta}) \mid \boldsymbol{x}_j \in \boldsymbol{X}_j, \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$, where $\boldsymbol{x}(t; t_j, \boldsymbol{x}_j, \boldsymbol{\theta})$ denotes a solution of $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta})$ for the initial condition $\boldsymbol{x} = \boldsymbol{x}_j$ at t_j. We will describe a method for determining enclosures $\boldsymbol{X}_j$ of the state variables at each time step $j = 1, \ldots, N$, such that $\boldsymbol{x}(t_j; t_0, \boldsymbol{X}_0, \boldsymbol{\Theta}) \subseteq \boldsymbol{X}_j$.

Assume that at t_j we have an enclosure $\boldsymbol{X}_j$ of $\boldsymbol{x}(t_j; t_0, \boldsymbol{X}_0, \boldsymbol{\Theta})$, and that we want to carry out an integration step to compute the next enclosure $\boldsymbol{X}_{j+1}$. Then, in the first phase of the method, the goal is to find a step size $h_j = t_{j+1} - t_j > 0$ and an a priori enclosure (coarse enclosure) $\widetilde{\boldsymbol{X}}_j$ of the solution such that a unique solution $\boldsymbol{x}(t; t_j, \boldsymbol{x}_j, \boldsymbol{\theta}) \in \widetilde{\boldsymbol{X}}_j$ is guaranteed to exist for all $t \in [t_j, t_{j+1}]$, all $\boldsymbol{x}_j \in \boldsymbol{X}_j$, and all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. We apply a traditional interval method, with high order enclosure, to the parametric ODEs by using an interval Taylor series (ITS) with respect to time. That is, we determine h_j and $\widetilde{\boldsymbol{X}}_j$ such that for $\boldsymbol{X}_j \subseteq \widetilde{\boldsymbol{X}}_j^0$,

$$\widetilde{\boldsymbol{X}}_j = \sum_{i=0}^{k-1} [0, h_j]^i \boldsymbol{F}^{[i]}(\boldsymbol{X}_j, \boldsymbol{\Theta}) + [0, h_j]^k \boldsymbol{F}^{[k]}(\widetilde{\boldsymbol{X}}_j^0, \boldsymbol{\Theta}) \subseteq \widetilde{\boldsymbol{X}}_j^0. \tag{10}$$

Here $\widetilde{\boldsymbol{X}}_j^0$ is an initial estimate of $\widetilde{\boldsymbol{X}}_j$, k denotes the order of the Taylor expansion, and the coefficients $\boldsymbol{F}^{[i]}$ are interval extensions of the Taylor coefficients $\boldsymbol{f}^{[i]}$ of $\boldsymbol{x}(t)$ with respect to time, which can be obtained recursively in terms of $\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta})$ by

$$\begin{aligned} \boldsymbol{f}^{[0]} &= \boldsymbol{x} \\ \boldsymbol{f}^{[1]} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}) \\ \boldsymbol{f}^{[i]} &= \frac{1}{i} \left(\frac{\partial \boldsymbol{f}^{[i-1]}}{\partial \boldsymbol{x}} \boldsymbol{f} \right) (\boldsymbol{x}, \boldsymbol{\theta}), \quad i \geq 2. \end{aligned} \tag{11}$$

Satisfaction of eq 10 demonstrates that there exists a unique solution $\boldsymbol{x}(t; t_j, \boldsymbol{x}_j, \boldsymbol{\theta}) \in \widetilde{\boldsymbol{X}}_j$ for all $t \in [t_j, t_{j+1}]$, all $\boldsymbol{x}_j \in \boldsymbol{X}_j$, and all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.

In the second phase of the method, we compute a tighter enclosure $\boldsymbol{X}_{j+1} \subseteq \widetilde{\boldsymbol{X}}_j$, such that $\boldsymbol{x}(t_{j+1}; t_0, \boldsymbol{X}_0, \boldsymbol{\Theta}) \subseteq \boldsymbol{X}_{j+1}$. This will be done by using an ITS approach to compute a Taylor model $\boldsymbol{T}_{\boldsymbol{x}_{j+1}}$ of $\boldsymbol{x}_{j+1}$ in terms of the initial values and parameters, and then obtaining the enclosure $\boldsymbol{X}_{j+1} = B(\boldsymbol{T}_{\boldsymbol{x}_{j+1}})$. For the Taylor model computations, we begin by representing the interval initial states and parameters by the Taylor models $\boldsymbol{T}_{\boldsymbol{x}_0}$ and $\boldsymbol{T}_{\boldsymbol{\theta}}$, respectively, with components

$$T_{x_{i0}} = (m(X_{i0}) + (x_{i0} - m(X_{i0})), [0, 0]), \quad i = 1, \cdots, m, \tag{12}$$

and

$$T_{\theta_i} = (m(\Theta_i) + (\theta_i - m(\Theta_i)), [0, 0]), \quad i = 1, \cdots, p. \tag{13}$$

Then, we can determine Taylor models $\boldsymbol{T}_{\boldsymbol{f}^{[i]}}$ of the interval Taylor series coefficients $\boldsymbol{f}^{[i]}(\boldsymbol{x}_i, \boldsymbol{\theta})$ by using RDA operations to compute $\boldsymbol{T}_{\boldsymbol{f}^{[i]}} = \boldsymbol{f}^{[i]}(\boldsymbol{T}_{\boldsymbol{x}_j}, \boldsymbol{T}_{\boldsymbol{\theta}})$. Using an interval Taylor series for $\boldsymbol{x}_{j+1}$ with coefficients given by $\boldsymbol{T}_{\boldsymbol{f}^{[i]}}$, and incorporating a novel approach for using the mean value theorem on Taylor models, one can obtain a result for $\boldsymbol{T}_{\boldsymbol{x}_{j+1}}$ in terms of the parameters and initial states. In order to address the wrapping effect,[6] results are propagated from one time step to the next using a new type of Taylor model, in which the remainder bound is not an interval, but a parallelepiped. That is, the remainder bound is a set of the form $\mathcal{P} = \{Av \mid v \in V\}$, where $A \in \mathbb{R}^{n \times n}$ is a real and regular matrix. If A is orthogonal, as from a QR-factorization, then $\mathcal{P}$ can be interpreted as a rotated n-dimensional rectangle. Complete details of the computation of $\boldsymbol{T}_{\boldsymbol{x}_{j+1}}$ are given by Lin and Stadtherr.[14]

The approach outlined above, as implemented in VSPODE, has been tested by Lin and Stadtherr,[14] who compared its performance with results obtained using the popular VNODE package[12] (in using VNODE, interval-valued parameters are treated as additional state variables with interval-valued initial states). In these tests, VSPODE was able to provide comparable or better (tighter) enclosures of the state variables than VNODE, while requiring an order of magnitude less computation time.

5 Guaranteed Nonlinear State and Parameter Estimator

The proposed state and parameter estimator is based on a type of "predictor-corrector" approach. The prediction step aims at computing the attainable set for the state vector, while the correction step retains only the parts of the set which are compatible with the bounded-error measurements. This is combined with a subinterval reconstitution scheme to allow handling of relatively large intervals of uncertainty and to improve the resolution of the results. In such a scheme, intervals of uncertainty are subdivided and each resulting subinterval is processed independently using the predictor-corrector procedure. The final results are then reconstituted simply by taking the union of the results from each subinterval.

The prediction step begins with a box $\boldsymbol{\Theta}$ that is known to contain the uncertain parameter values, and a box $\boldsymbol{X}_0$ that is known to contain the uncertain initial states. Using a validated solver for parametric ODEs, an outer approximation of the predicted attainable set $\boldsymbol{X}_j^+$ at t_j can be computed. When the VSPODE package[14] is used, the set is represented by the Taylor model $\boldsymbol{T}_{\boldsymbol{x}_j}$ in terms of the uncertain quantities (initial values $\boldsymbol{x}_0 \in \boldsymbol{X}_0$ and parameters $\boldsymbol{\theta} \in \boldsymbol{\Theta}$). This set is guaranteed to contain the true state at t_j, i.e. $\boldsymbol{X}_j \subseteq \boldsymbol{X}_j^+ = B(\boldsymbol{T}_{\boldsymbol{x}_j})$.

In the correction step, a Taylor model of the output variables, $\boldsymbol{T}_{\boldsymbol{y}_j}$, is calculated from $\boldsymbol{y}_j = \boldsymbol{g}(\boldsymbol{x}_j, \boldsymbol{\theta})$ using $\boldsymbol{T}_{\boldsymbol{x}_j}, \boldsymbol{T}_{\boldsymbol{\theta}}$, and RDA operations. That is $\boldsymbol{T}_{\boldsymbol{y}_j} = \boldsymbol{g}(\boldsymbol{T}_{\boldsymbol{x}_j}, \boldsymbol{T}_{\boldsymbol{\theta}})$, a Taylor model in terms of the uncertain initial states $\boldsymbol{x}_0$ and parameters $\boldsymbol{\theta}$. This must be compatible with the bounded-error measurements, that is, the bound constraint $\boldsymbol{y}_j \in \boldsymbol{Y}_j = \widehat{\boldsymbol{y}}_j - \boldsymbol{V}_j$ must be satisfied. In the next subsection we describe a procedure for using this constraint to eliminate parts of $\boldsymbol{\Theta}$ and $\boldsymbol{X}_0$ which are incompatible with the measurements. Then, using these updated intervals for the uncertain quantities, the estimates $\boldsymbol{X}_j = B(\boldsymbol{T}_{\boldsymbol{x}_j})$ for the state vector can be obtained.

5.1 Constraint Propagation Procedure

Information expressed by a constraint can be used to eliminate incompatible values from the domain of its variables. The domain reduction can then be propagated to all constraints on that variable, which may be used to further reduce the domains of other variables. This process is

known as constraint propagation,[2,19] and is widely used in various forms (e.g., hull consistency) in connection with interval methods. In this subsection, we show how to use Taylor models to apply such a constraint propagation procedure for bound constraints.

Let T_c be the Taylor model of the function $c(\boldsymbol{x})$ over the interval $\boldsymbol{x} \in \boldsymbol{X}$, and say the bound constraint $\underline{R} \le c(\boldsymbol{x}) \le \overline{R}$ must be satisfied; that is, $c(\boldsymbol{x}) \in R$, where $R = [\underline{R}, \overline{R}]$ is some specified interval. This is equivalent to the constraint $c(\boldsymbol{x}) - r = 0$ with $r \in R$. We seek to eliminate parts of $\boldsymbol{X}$ for which it can be guaranteed that the constraint will not be satisfied. In the constraint propagation procedure (CPP) described here, $B(T_c)$ is first determined, thus bounding $c(\boldsymbol{x}) - r$, $r \in R$, by $B(T_c) - R$. There are now two possible outcomes: 1. If $\overline{B(T_c) - R} < 0$ or $\underline{B(T_c) - R} > 0$, that is $0 \notin B(T_c) - R$, then no $\boldsymbol{x} \in \boldsymbol{X}$ will ever satisfy the constraint; thus, the CPP can be stopped and $\boldsymbol{X}$ discarded. 2. Otherwise, if $0 \in B(T_c) - R$, then it still may be possible to eliminate at least *part* of $\boldsymbol{X}$; thus the CPP continues as described below, using an approach based on the bounding strategy for Taylor models described in Section 3.2.

For some component i of $\boldsymbol{x}$, let a_i and b_i be the polynomial coefficients of the terms $(x_i - x_{i0})^2$ and $(x_i - x_{i0})$ of T_c, respectively. Note that, $x_{i0} \in X_i$ and is usually the midpoint $x_{i0} = m(X_i)$; the value of x_{i0} will not change during the CPP. For $|a_i| \ge \omega$, the bound of T_c can be expressed using eq 8 as

$$B(T_c) = a_i \left(X_i - x_{i0} + \frac{b_i}{2a_i} \right)^2 - \frac{b_i^2}{4a_i} + S_i. \tag{14}$$

where

$$S_i = \sum_{\substack{j=1 \\ j \ne i}}^{m} \left[a_j \left(X_j - x_{j0} + \frac{b_j}{2a_j} \right)^2 - \frac{b_j^2}{4a_j} \right] + Q. \tag{15}$$

We can reduce the computational effort to obtain S_i by recognizing that this quantity is just $B(T_c)$ less the i-th term in the summation, and $B(T_c)$ was already computed earlier in the CPP. Thus, for each i, S_i can be determined by dependent subtraction (see above) using

$$S_i = B(T_c) \ominus \left[a_i \left(X_i - x_{i0} + \frac{b_i}{2a_i} \right)^2 - \frac{b_i^2}{4a_i} \right]. \tag{16}$$

Now define the intervals $U_i = X_i - x_{i0} + \frac{b_i}{2a_i}$ and $V_i = \frac{b_i^2}{4a_i} - S_i$, so that $B(T_c) = a_i U_i^2 - V_i$. The goal is to identify and retain only the part of X_i that contains values of x_i for which it is

possible to satisfy the constraint $c(\boldsymbol{x}) - r = 0$, $r \in R$. Thus, the part of X_i that is going to be eliminated is guaranteed not to satisfy the constraint. The range of $c(\boldsymbol{x}) - r$, $r \in R$, is bounded by $B(T_c) - R = a_i U_i^2 - V_i - R = a_i U_i^2 - (V_i + R)$. Thus, the part of X_i in which it is possible to satisfy $c(\boldsymbol{x}) - r = 0$, $r \in R$, can be bounded by finding X_i such that all elements of $a_i U_i^2$ are in $V_i + R$. This corresponds, with $W_i = (V_i + R)/a_i$, to the requirement that

$$\overline{U_i^2} \leq \overline{W_i} \text{ and } \underline{U_i^2} \geq \underline{W_i} \tag{17}$$

Then, the set U_i that satisfies eq 17 can be determined to be

$$U_i = \begin{cases} \emptyset & \text{if } \overline{W_i} < 0 \\ \left[-\sqrt{\overline{W_i}}, \sqrt{\overline{W_i}}\right] & \text{if } \underline{W_i} \leq 0 \leq \overline{W_i} \\ -\sqrt{W_i} \cup \sqrt{W_i} & \text{if } \underline{W_i} > 0 \end{cases}. \tag{18}$$

Thus, the part of X_i retained is $X_i = X_i \cap \left(U_i + x_{i0} - \frac{b_i}{2a_i}\right)$.

If $|a_i| < \omega$ and $|b_i| \geq \omega$, then eq 8 should not be used, but eq 7 can be used instead. Following a procedure similar to that used above, we now have $B(T_c) = b_i U_i - V_i$ with $U_i = X_i - x_{i0}$ and $V_i = -\left(B(T_c) \ominus b_i(X_i - x_{i0})\right)$. Note that all quadratic terms are now included in V_i. To identify bounds on the part of X_i in which it is possible to satisfy the constraint, the set U_i can be determined to be $U_i = (V_i + R)/b_i$. Thus, the part of X_i retained is $X_i = X_i \cap (U_i + x_{i0})$. If both $|a_i|$ and $|b_i|$ are less than ω, then no CPP will be applied on X_i.

The overall CPP is implemented by beginning with $i = 1$ and proceeding component by component. If, for any i, the result $X_i = \emptyset$ is obtained, then no $\boldsymbol{x} \in \boldsymbol{X}$ can satisfy the constraint; thus, $\boldsymbol{X}$ can be discarded and the CPP stopped. Otherwise the CPP proceeds until all components of $\boldsymbol{X}$ have been updated. Note that, in principle, each time an improved (smaller) X_i is found, it could be used in computing S_i for subsequent components of $\boldsymbol{X}$. However, this requires recomputing the bound $B(T_c)$, which, for the functions $c(\boldsymbol{x})$ that will be of interest here, is expensive. Thus, the CPP for each component is done using the bounds $B(T_c)$ computed from the original $\boldsymbol{X}$. If, after each component is processed, $\boldsymbol{X}$ has been sufficiently reduced (by more than 10% by volume), then a new bound $B(T_c)$ is obtained, now over the smaller $\boldsymbol{X}$, and a new CPP is started. Otherwise,

the CPP terminates.

5.2 Estimation Algorithm

The core of the estimation algorithm is the prediction-correction procedure that is given as Algorithm 1. The overall estimation algorithm, including the subinterval reconstitution scheme, is then given as Algorithm 2. In both procedures, it is sometimes convenient to treat the uncertain quantities (initial states and parameters) as a single vector $\boldsymbol{z} = (\boldsymbol{\theta}, \boldsymbol{x}_0)^{\mathrm{T}} \in \boldsymbol{Z} = (\boldsymbol{\Theta}, \boldsymbol{X}_0)^{\mathrm{T}}$.

In Algorithm 1 (Prediction_Correction), the inputs are the interval vector of uncertain quantities $\boldsymbol{Z}$, the set of measurement times $\boldsymbol{t} = \{t_j, j = 1, \ldots, N\}$, the set of bounded-error measurements $\boldsymbol{Y} = \{\boldsymbol{Y}_j, j = 1, \ldots, N\}$, and the number of measurement times N. The outputs are the updated interval vector $\boldsymbol{Z}$, the set of state enclosures $\boldsymbol{X} = \{\boldsymbol{X}_j, j = 1, \ldots, N\}$, and a return flag. On Line 1, the procedure Taylor_init refers to the application of eqs 12 and 13 to initialize the Taylor models used. If the input interval $\boldsymbol{Z}$ of uncertain quantities is too large, then it is possible that, at some time instance j, VSPODE may fail to verify and bound the solution of the ODE system, thus causing a result of info = .false. on Line 3. Thus, no meaningful solution enclosure can be obtained, and Algorithm 1 returns FAIL on Line 12. This issue will be dealt with by subdividing $\boldsymbol{Z}$ in Algorithm 2. The procedure Bound_Constraint_Propagation (Line 6) carries out the constraint propagation procedure (CPP) for Taylor models, as described in Section 5.1. On Line 7, if the interval returned from the CPP is empty, it means that the input domain $\boldsymbol{Z} = (\boldsymbol{\Theta}, \boldsymbol{X}_0)^{\mathrm{T}}$ for the uncertain quantities is not compatible with the measurements; thus it can be discarded and the algorithm returns INCOMPATIBLE. If the input domain contains values of $\boldsymbol{z}$ that are compatible with the measurements and other values that are incompatible, then the CPP may discard parts of the domain that are incompatible, but will retain all compatible values. This possesses a significant advantage over the approaches of Jaulin[1] and Raïssi et al.,[8] in which only a domain that is wholly incompatible can be discarded. On Line 16, if the input domain interval has been tested through all the time instances, the state enclosures will be computed and the procedure will return SUCCESS.

If the given input interval $\boldsymbol{Z} = (\boldsymbol{\Theta}, \boldsymbol{X}_0)^{\mathrm{T}}$ of uncertainties is too large, the prediction and correction procedure (Algorithm 1) may return FAIL if VSPODE fails to verify and enclose the

solution to the ODE system. In addition, even in the case that the return from the prediction and correction procedure is SUCCESS, it sometimes may be desirable to obtain higher resolution in the results by using smaller input intervals. To deal with these cases, the input interval $\boldsymbol{Z}$ can be divided into smaller subintervals, which can then be tested one at a time in Algorithm 1. The overall results are then reconstituted by combining the results from each subinterval. To do this, the iteration procedure summarized in Algorithm 2 (Estimator) is used, based on a specified volume reduction fraction Ω and on specified tolerance vectors $\boldsymbol{\epsilon}_Z$ and $\boldsymbol{\epsilon}_X$. The algorithm output $\mathcal{L}_r$ is a list that contains parameter and state enclosures that are consistent with the measured outputs. All values consistent with the measurements are guaranteed to be enclosed.

Algorithm 2 tests a series of subintervals using the prediction and correction procedure (Algorithm 1) on each, with the list $\mathcal{L}_t$ containing subintervals remaining to be tested. The possible return values from Algorithm 1 (Prediction_Correction procedure on Line 4) are INCOMPATIABLE, SUCCESS, or FAIL. If the return from the prediction and correction procedure is INCOMPATIBLE (Line 5), then processing of the current subinterval will be stopped (current = .false.) and this subinterval will be discarded. If the current subinterval has been significantly reduced (as specified by the Ω input), its processing will continue (current = .true.) and it will be tested again without bisection (Lines 8 and 9). On Line 10, when the flag bisection_condition is true, a bisection operation will be performed. This results in the current subinterval being bisected into two subintervals, one of which will be stored in the testing list $\mathcal{L}_t$, with the other to be tested at the next iteration. The flag bisection_condition is true when: 1. The return from the prediction and correction procedure is FAIL and the size of the current interval $\boldsymbol{Z}$ is larger than the tolerance $\boldsymbol{\epsilon}_Z$, or 2. Further refinement of the solution enclosure is desired, as indicated by the size of the current interval $\boldsymbol{Z}$ and/or the solution enclosure $\boldsymbol{X}$ being larger than the corresponding tolerance $\boldsymbol{\epsilon}_Z$ and/or $\boldsymbol{\epsilon}_X$. Note that the components of the tolerance vectors $\boldsymbol{\epsilon}_Z$ and $\boldsymbol{\epsilon}_X$ may be different for each component of $\boldsymbol{z}$ and $\boldsymbol{x}$, and that interval size is determined on a componentwise basis. That is, only when the size (diameter) of each component of $\boldsymbol{Z}$ and/or $\boldsymbol{X}$ is less than the corresponding tolerance is the subinterval determined to be sufficiently small to avoid bisection. If the subinterval is sufficiently small and has been successfully tested in the prediction and correction procedure, it, along with

the state enclosure at each sampling instance, is then stored in the output list (Lines 14 and 15). If the current subinterval is sufficiently small and VSPODE has failed to verify the solution, the procedure will abort and return FAIL (Line 18), which suggests that a reduced size tolerance must be required for one or more component. In a successful return, the parameter and state enclosures from each tested subinterval are stored in the output list $\mathcal{L}_r$. The final overall parameter and state enclosures are then reconstituted by taking the union of the enclosures stored in the output list. If desired, the final overall enclosures may also be represented by their interval hull. If the output list is empty upon return, it means that there are no initial state and/or parameter values in the initially specified $\boldsymbol{X}_0$ and $\boldsymbol{\Theta}$ that are consistent with the measurements.

6 Computational Studies

In this section, we present the results of numerical experiments using three different models. The numerical experiments were performed on a workstation running Linux with an Intel Pentium 4 3.2GHz CPU. Results from VSPODE were obtained using a $k = 11$ order interval Taylor QR method, and with a $q = 3$ order Taylor model. The volume reduction threshold $\Omega = 0.75$ was used. The size tolerances used are given in each subsection below. Any size tolerances not given were set to some arbitrarily large number, indicating no size tolerance was enforced for these quantities.

6.1 Lotka-Volterra Model

We consider the Lotka-Volterra predator-prey model, which describes the population dynamics of two species, with biomasses represented by x_1 and x_2. The model is represented by the following differential equations:

$$\dot{x}_1 = (a - bx_2)x_1 \tag{19}$$

$$\dot{x}_2 = (dx_1 - c)x_2, \tag{20}$$

where a, b, c, and d are positive parameters. The outputs are $y_1 = x_1$ and $y_2 = x_2$. Output measurements were simulated by solving the model from $t = 0$ to $t = 10$ with constant time

step $h = 0.01$, using the initial state $\boldsymbol{x}_0 = (50, 50)^{\mathrm{T}}$ and known parameters $\boldsymbol{\theta} = (a, b, c, d)^{\mathrm{T}} = (1, 0.01, 1, 0.02)^{\mathrm{T}}$, and adding uniform noise in the interval $V_j = [-1, 1]$ at each measurement time t_j. The size tolerances used are $\epsilon = 1$ and $\epsilon = 0.001$ for initial states and parameters, respectively. In the following three numerical experiments, we will estimate 1) state, 2) parameters and 3) state and parameters simultaneously.

6.1.1 State Estimation

In this experiment, we assume that bounded-error measurements for only one output ($y_1 = x_1$) are available, and that the parameters are known exactly (as specified above). The goal then is to estimate the unmeasured state variable x_2. The initial value of x_2 was assumed to be in the interval $[0, 100]$. The initial value of x_1 was assumed to be in the interval $[49, 51]$, to be consistent with the measurement error bounds. In a computation time of 12.9 seconds, the estimation procedure described above generated the enclosure of x_2 depicted in Fig. 1. The enclosure of x_2 at $t = 0$ has been reduced dramatically, to $[49.1464, 50.8707]$, compared to the assumed interval $[0, 100]$. The majority of the assumed initial interval has been proved incompatible with the measurements, and thus has been deleted during the state estimation procedure. As shown in Fig. 1, a good enclosure of x_2 was obtained from $t = 0$ to $t = 10$. The enclosure of x_2 at $t = 10$ is $[148.9294, 154.5899]$.

6.1.2 Parameter Estimation

In this experiment, the bounded error measurements of both outputs ($y_1 = x_1$ and $y_2 = x_2$) are available, and the goal is to estimate the parameters b and d, with the other parameters assumed known exactly (at the values specified above). The initial intervals assumed for the two parameters are $b \in [0, 1]$ and $d \in [0, 1]$. The initial values of both states were assumed to be in the interval $[49, 51]$, for consistency with the measurement error bounds. In a computation time of 35.5 seconds, the estimation procedure generated the parameter enclosure shown in Fig. 2, which shows the union of the four enclosures in the final output list $\mathcal{L}_r$. Using the interval hull of this enclosure gives $b \in [0.009883, 0.010111]$ and $d \in [0.019626, 0.0203934]$ for the parameters, as compared to their true values of $b = 0.01$ and $d = 0.02$.

A similar parameter estimation example with uncertain initial values was studied by Raïssi et al.[8] In this case, bounded-error measurement data for only one output ($y_1 = x_1$) is available. The measurements were simulated as described above, but with uniform noise in the interval $V_j = [-1.5, 1.5]$ and spanning $t = 0$ to $t = 7$ with constant time step $h = 0.005$. Again, b and d are to be estimated, with the other parameters assumed known. Raïssi et al.[8] assumed the initial values of both states to be in the interval $[49, 51]$. This is inconsistent with the available measurement data, since there are no measurements for x_2, and x_1 is measured with an error of ± 1.5. Nevertheless, for the sake of comparison, we have tested the estimation procedure described above on this same problem. In 439.5 seconds of computation time, the estimation procedure generated the enclosure of the parameters depicted in Fig 3, which shows all of the parameter boxes in the output list $\mathcal{L}_r$. The interval hull of this enclosure is $b \in [0.009271, 0.01085]$ and $d \in [0.01954, 0.02051]$, representing a maximum absolute error of 8.5% in b and 2.6% in d. Raïssi et al.[8] reported obtaining a maximum absolute error of 14% in b and 7.7% in d, corresponding to an enclosure with an interval hull of about $b \in [0.0087, 0.0114]$ and $d \in [0.0185, 0.0215]$ (approximate values obtained from Raïssi et al.'s Fig. 4), in a computation time of 1 hour on a Pentium 4 1.6 GHz machine. Considering that the machine used in our experiments is about twice as fast, it appears that the approach presented here can obtain a better enclosure, and in about a fourth of the computation time.

6.1.3 State and Parameter Estimation

In this final experiment with the Lotka-Volterra model, bounded-error measurements of one output ($y_1 = x_1$) are available, and the goal is to simultaneously estimate one unknown state variable (x_2) and one unknown parameter (d), with the other parameters fixed at the values given above. The initial interval for d was assumed to be $[0, 1]$, and the initial value of x_2 was assumed to be in the interval $[0, 100]$. The initial value of x_1 was assumed to be in the interval $[49, 51]$, which is consistent with the measurement error bounds. In a computation time of 115 seconds, the estimation procedure generated an enclosure for d of $[0.019647, 0.0203704]$ and the enclosure of x_2 depicted in Fig. 4. The enclosure of x_2 at $t = 0$ is again reduced dramatically, to $[48.2580, 51.6546]$,

and a good state enclosure is obtained from $t = 0$ to $t = 10$, with a final enclosure at $t = 10$ of $[146.2092, 157.4294]$.

6.2 Two-state Microbial Growth Model

We consider a simple microbial growth process[31] in a bioreactor, which involves a single biomass and a single substrate. The process can be described using the following ODE model:

$$\dot{X} = (\mu - \alpha D)X \tag{21}$$

$$\dot{S} = D(S^i - S) - k\mu X, \tag{22}$$

where X and S are concentrations of biomass and substrate, respectively; α is the process heterogeneity parameter; D and S^i are the dilution rate and the influent concentration of substrate, respectively; k is the yield coefficient; and μ is the growth rate, which is dependent on S. In this study, we use the Haldane law for the growth rate,

$$\mu = \frac{\mu_m S}{K_S + S + K_I S^2}\ , \tag{23}$$

where μ_m is the maximum growth rate, K_S is the saturation parameter, and K_I is the inhibition parameter. Here, the initial values of the biomass concentration X_0, and the process kinetic parameters (μ_m, K_S, and K_I) are assumed to be uncertain and given by intervals. The inputs, D and S^i, and other parameters, α and k, are assumed known, with values as shown in Table 1. The only output measurement available is the concentration of substrate ($y = S$). The output data $\widehat{y}_j = \widehat{S}_j, j = 1, \ldots N$, were simulated by solving the model from $t = 0$ to $t = 20$ days with constant time step $h = 0.2$, and using the initial state $(X_0, S_0) = (0.83, 0.80)$ with assumed parameter values $(\mu_m, K_S, K_I) = (1.2, 7.1, 1/256)$, and adding noise that is $\pm 1\%$ of the measurement. That is, the measurement error bounds are given by $Y_j = \widehat{y}_j \times [0.99, 1.01]$ at each time step t_j. The goal of the experiment is to estimate the unknown state variable, namely the biomass concentration X, from $t = 0$ to $t = 20$ days, and to estimate the uncertain parameters, μ_m, K_S, and K_I. The intervals initially assumed for the initial conditions (X_0, S_0) and for the uncertain parameters (μ_m, K_S, and K_I) are given in Table 1. The only size tolerance (stopping criterion) is that the width of subinterval enclosures of X at $t = 20$ be less than $\epsilon = 0.01$ g/L.

After 719.7 seconds computation time, the estimation procedure generated the enclosure of biomass concentration (X) depicted in Fig. 5. The interval of the initial value X_0 was reduced to $[0.7944, 0.8640]$ g/L from its assumed value of $[0.4, 1.2]$ g/L. A good enclosure of X from $t = 0$ to $t = 20$ was obtained, with a final enclosure at $t = 20$ of $[0.8358, 0.8476]$ g/L. The enclosure of the parameters, μ_m and K_S is shown in Fig. 6, where all the boxes in the output list $\mathcal{L}_r$ are depicted. The interval hull of μ_m was reduced somewhat to $[1.0316, 1.3445]$ day^{-1}, but there was no reduction of the interval hull for the other two parameters. However, a clear relationship between μ_m and K_S has been identified, suggesting some correlation of these parameters, at least for this error-bounded data set in which only the substrate concentration is measured.

6.3 Three-state Biochemical Reactor

We consider a biochemical reactor[32] where the consumption of substrate (x_2) promotes the growth of biomass (x_1) and formation of product (x_3). The process can be described using the following ODE model:

$$\dot{x}_1 = (\mu - D)x_1 \tag{24}$$

$$\dot{x}_2 = D(x_{2f} - x_2) - \frac{\mu x_1}{Y} \tag{25}$$

$$\dot{x}_3 = -Dx_3 + (\alpha\mu + \beta)x_1 \tag{26}$$

where the specific growth rate μ is a function of both the substrate and the product concentrations:

$$\mu = \frac{\mu_m(1 - x_3/x_{3m})x_2}{k_s + x_2}. \tag{27}$$

Here, we assume the maximum growth rate μ_m, and the saturation parameter k_s are unknown, but belong to intervals of $[0.2, 0.6]$ hr^{-1} and $[1, 2]$ g/L, respectively. The biomass concentrations are unknown but within the interval of $[2, 10]$ g/L. Other parameters and inputs are assumed known exactly with values shown in Table 2. The output measurements of substrate and product concentrations are available, but subject to measurement noise of ± 0.01 g/L. The output data $\widehat{\boldsymbol{y}}_j = (\widehat{x}_{2,j}, \widehat{x}_{3,j})^{\mathrm{T}}$, $j = 1, \ldots N$, were simulated by solving the model from $t = 0$ to $t = 20$ hours with constant time step $h = 0.2$, using the initial state of $\boldsymbol{x}_0 = (6.5, 5, 15)^{\mathrm{T}}$ and the assumed

parameter values $(\mu_m, k_s) = (0.48, 1.2)$, and adding measurement noise. Then, the measurement error bounds are given by $\boldsymbol{Y}_j = \widehat{\boldsymbol{y}}_j + [-0.01, 0.01]$ at each time step t_j. The goal of the experiment is to estimate the unknown state variable, namely the biomass concentrations x_1, from $t = 0$ to $t = 20$ hours, and to estimate the uncertain kinetic parameters, μ_m and k_s. The only size tolerance (stopping criterion) is that the width of subinterval enclosures of x_1 at $t = 20$ be less than $\epsilon = 0.01$ g/L.

After 1450.8 seconds computation time, the estimation procedure generated the enclosure of biomass concentrations (x_1) depicted in Fig. 7. The interval of the initial value was reduced to $[6.4549, 6.5676]$ g/L from its assumed value of $[2, 10]$ g/L. A good enclosure of x_1 from $t = 0$ to $t = 20$ hours was obtained, with a final enclosure at $t = 20$ of $[6.6375, 6.7581]$ g/L. The enclosure of the kinetic parameters, μ_m and k_s is shown in Fig. 8, where all the boxes in the output list $\mathcal{L}_r$ are depicted. The interval hull of μ_m was reduced to $[0.4595, 0.5004]$ hr^{-1} from $[0.2, 0.6]$ hr^{-1} and the interval hull of k_s was reduced to $[1.0344, 1.3750]$ g/L from $[1, 2]$ g/L. A clear relationship between μ_m and k_s has also been identified, suggesting some correlation of these parameters.

7 Concluding Remarks

We have described here a technique for state and parameter estimation in nonlinear, continuous-time systems. The method provides guaranteed enclosures of all state and parameter values that are consistent with bounded-error output measurements. The technique is based on the use of interval analysis and features: 1. Use of a new validated solver for parametric ODEs, which is used to produce guaranteed bounds on the solutions of nonlinear dynamic systems with interval-valued parameters and initial states; 2. Use of a constraint propagation strategy on the Taylor models used to represent the solutions of the dynamic system; and 3. Use of a subinterval reconstitution scheme to permit handling of relatively large intervals of uncertainty and refinement of results. Numerical experiments have demonstrated the usefulness of the method, as well as its computational efficiency.

Acknowledgment

This work was supported in part by the State of Indiana 21st Century Research and Technology Fund under Grant #909010455, and by the Department of Energy under Grant DE-FG02-05CH11294.

References

(1) Jaulin, L. Nonlinear bounded-error state estimation of continuous-time systems. *Automatica* **2002**, *38*, 1079–1082.

(2) Jaulin, L.; Kieffer, M.; Didrit, O.; É Walter, *Applied Interval Analysis*; Springer-Verlag: London, UK, 2001.

(3) Chernousko, F. L. *State Estimation for Dynamic System*; CRC Press: Boca Raton, FL, 1994.

(4) Chen, G.; Wang, J.; Shieh, L. S. Interval Kalman filtering. *IEEE T. Aero. Elec. Sys.* **1997**, *33*, 250–258.

(5) Schweppe, F. C. Recursive state estimation: Unknown but bounded errors and system inputs. *IEEE T. Automat. Contr.* **1968**, *13*, 22–28.

(6) Moore, R. E. *Interval Analysis*; Prentice-Hall: Englewood Cliffs, NJ, 1966.

(7) Davis, E. Constraint propagation with interval labels. *Artif. Intell.* **1987**, *32*, 281–331.

(8) Raïssi, T.; Ramdani, N.; Candau, Y. Set membership state and parameter estimation for systems described by nonlinear differential equations. *Automatica* **2004**, *40*, 1771–1777.

(9) Rihm, R. Interval methods for initial value problems in ODEs. In *Topics in Validated Computations: Proceedings of the IMACS-GAMM International Workshop on Validated Computations*; Herzberger, J., Ed.; Elsevier: New York, NY, 1994.

(10) Nedialkov, N. S.; Jackson, K. R.; Corliss, G. F. Validated solutions of initial value problems for ordinary differential equations. *Appl. Math. Comput.* **1999**, *105*, 21–68.

(11) Lohner, R. J. Computations of guaranteed enclosures for the solutions of ordinary initial and boundary value problems. In *Computational Ordinary Differential Equations*; Cash, J.; Gladwell, I., Eds.; Clarendon Press: Oxford, UK, 1992.

(12) Nedialkov, N. S.; Jackson, K. R.; Pryce, J. D. An effective high-order interval method for validating existence and uniqueness of the solution of an IVP for an ODE. *Reliab. Comput.* **2001**, *7*, 449–465.

(13) Berz, M.; Makino, K. Verified integration of ODEs and flows using differential algebraic methods on high-order Taylor models. *Reliab. Comput.* **1998**, *4*, 361–369.

(14) Lin, Y.; Stadtherr, M. A. Validated solutions of initial value problems for parametric ODEs. *Appl. Num. Math.* **2007**, published online as dx.doi.org/10.1016/j.apnum.2006.10.006, in press.

(15) Makino, K.; Berz, M. Remainder differential algebras and their applications. In *Computational Differentiation: Techniques, Applications, and Tools*; Berz, M.; Bishof, C.; Corliss, G.; Griewank, A., Eds.; SIAM: Philadelphia, PA, 1996.

(16) Makino, K.; Berz, M. Efficient control of the dependency problem based on Taylor model methods. *Reliab. Comput.* **1999**, *5*, 3–12.

(17) Lin, Y.; Stadtherr, M. A. Validated solution of initial value problems for ODEs with interval parameters. Presented at: 2nd NSF Workshop on Reliable Engineering Computing; Savannah, GA, February 22–24, 2006.

(18) Kletting, M.; Rauh, A.; Aschemann, H.; Hofer, E. P. Interval observer design based on Taylor models for nonlinear uncertain continuous-time systems. Presented at: 12th GAMM–IMACS International Symposion on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN 2006); Duisburg, Germany, September 26–29, 2006.

(19) Hansen, E.; Walster, G. W. *Global Optimization Using Interval Analysis*; Marcel Dekker: New York, NY, 2004.

(20) Pryce, J. D.; Corliss, G. F. Interval arithmetic with containment sets. *Computing* **2006**, *78*, 251–276.

(21) Kearfott, R. B. *Rigorous Global Search: Continuous Problems*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1996.

(22) Neumaier, A. *Interval Methods for Systems of Equations*; Cambridge University Press: Cambridge, UK, 1990.

(23) Ratschek, H.; Rokne, J. *Computer Methods for the Range of Functions*; Ellis Horwood: Chichester, UK, 1984.

(24) Makino, K. *Rigorous Analysis of Nonlinear Motion in Particle Accelerators,* PhD Thesis, Michigan State University, East Lansing, MI, 1998.

(25) Makino, K.; Berz, M. Taylor models and other validated functional inclusion methods. *Int. J. Pure Appl. Math.* **2003**, *4*, 379–456.

(26) Revol, N.; Makino, K.; Berz, M. Taylor models and floating-point arithmetic: Proof that arithmetic operations are validated in COSY. *J. Logic Algebr. Progr.* **2005**, *64*, 135–154.

(27) Neumaier, A. Taylor forms – Use and limits. *Reliab. Comput.* **2003**, *9*, 43–79.

(28) Makino, K.; Berz, M. Taylor model range bounding schemes. In *Third International Workshop on Taylor Methods*; Miami Beach, FL, 2004.

(29) Makino, K.; Berz, M. Verified global optimization with Taylor model-based range bounders. *Transactions on Computers* **2005**, *11*, 1611–1618.

(30) Eijgenraam, P. The Solution of Initial Value Problems Using Interval Arithmetic. Technical Report Mathematical Centre Tracts No. 144, Stichting Mathematisch Centrum, Amsterdam, The Netherlands, 1981.

(31) Bastin, G.; Dochain, D. *On-line Estimation and Adaptive Control of Bioreactors*; Elsevier, Amsterdam, The Netherlands: 1990.

(32) Bequette, B. W. *Process Dynamics: Modeling, Analysis, and Simulation*; Prentice-Hall, Upper Saddle River, NJ: 1998.

Algorithm 1 Prediction_Correction(in: $\boldsymbol{Z}, \boldsymbol{Y}, \boldsymbol{t}, N$; out: $\boldsymbol{Z}, \boldsymbol{X}$)

[$\boldsymbol{\Theta}, \boldsymbol{X}_0$] = $\boldsymbol{Z}$; Taylor_init($\boldsymbol{X}_0$;$\boldsymbol{T}_{\boldsymbol{x}_0}$); Taylor_init($\boldsymbol{\Theta}$;$\boldsymbol{T}_{\boldsymbol{\theta}}$); ret = SUCCESS
for $j = 0$ to $N - 1$ **do**
  info = VSPODE($\boldsymbol{T}_{\boldsymbol{\theta}}, \boldsymbol{T}_{\boldsymbol{x}_j}, t_j, t_{j+1}$; $\boldsymbol{T}_{\boldsymbol{x}_{j+1}}$)
  if info = .true. **then** {Successful return from VSPODE}
    $\boldsymbol{T}_{\boldsymbol{y}_{j+1}} = \boldsymbol{h}(\boldsymbol{T}_{\boldsymbol{x}_{j+1}}, \boldsymbol{T}_{\boldsymbol{\theta}})$
    Bound_Constraint_Propagation($\boldsymbol{T}_{\boldsymbol{y}_{j+1}}, \boldsymbol{Z}, \boldsymbol{Y}_{j+1}$; $\boldsymbol{Z}$)
    if $\boldsymbol{Z} = \emptyset$ **then** {Incompatible box}
      ret = INCOMPATIBLE
      break
    end if
  else {VSPODE fail}
    ret = FAIL
    break
  end if
end for
if ret = SUCCESS **then** {Complete prediction-correction procedure}
  $\boldsymbol{X}_j = \boldsymbol{B}(\boldsymbol{T}_{\boldsymbol{x}_j}), \quad j = 1, \ldots, N$
end if
return ret

Algorithm 2 Estimator(in: $\boldsymbol{\Theta}, \boldsymbol{X}_0, \boldsymbol{Y}, \boldsymbol{t}, N, \boldsymbol{\epsilon}_Z, \boldsymbol{\epsilon}_X, \Omega$; out: $\mathcal{L}_r$)

$\mathcal{L}_t = \emptyset$; $\mathcal{L}_r = \emptyset$; $\boldsymbol{Z} = [\boldsymbol{\Theta}, \boldsymbol{X}_0]$; current = .true.
while current = .true. **do**
  Vol_old = Volume($\boldsymbol{Z}$)
  info = Prediction_Correction($\boldsymbol{Z}, \boldsymbol{Y}, \boldsymbol{t}, N$; $\boldsymbol{Z}, \boldsymbol{X}$)
  if info = INCOMPATIBLE **then**
    current = .false.
  else {SUCCESS or FAIL}
    if Volume($\boldsymbol{Z}$)/Vol_old $< \Omega$ **then** {Box significantly reduced}
      current = .true.
    else if bisection_condition = .true. **then** {Do bisection}
      Bisect the box, resulting in $\boldsymbol{Z}$ and $\boldsymbol{Z}_1$
      Push $\boldsymbol{Z}_1$ into $\mathcal{L}_t$
      current = .true.
    else if info = SUCCESS **then** {Small box and SUCCESS}
      Push $[\boldsymbol{Z}, \boldsymbol{X}]$ into $\mathcal{L}_r$
      current = .false.
    else {Small box and FAIL}
      return FAIL
    end if
  end if
  if current = .false. **and** $\mathcal{L}_t \neq \emptyset$ **then**
    Pop one box from $\mathcal{L}_t$ as $\boldsymbol{Z}$
    current = .true.
  end if
end while
return SUCCESS

Table 1: Microbial growth model parameters

Parameter	Value	Units	Parameter	Value	Units
α	0.5	–	μ_m	[1.0, 1.4]	day^{-1}
k	10.53	g S/ g X	K_S	[6, 8]	g S/L
D	0.36	day^{-1}	K_I	[0.0025, 0.01]	$(\text{g S/L})^{-1}$
S^i	5.7	g S/L	X_0	[0.4, 1.2]	g X/L
			S_0	$0.80 \times [0.99, 1.01]$	g S/L

Table 2: Three-state biochemical reactor model parameters

Parameter	Value	Units	Parameter	Value	Units
Y	0.4	g/g	α	2.2	g/g
β	0.2	hr^{-1}	x_{3m}	50	g/L
D	0.202	hr^{-1}	x_{2f}	20	g/L

List of Figure Captions

Figure 1. Enclosure of unmeasured state variable x_2 in state estimation for the Lotka-Volterra model.

Figure 2. Enclosure of parameters b and d in parameter estimation for the Lotka-Volterra model.

Figure 3. Enclosure of parameters b and d in parameter estimation for the Lotka-Volterra model (using problem data given by Raïssi et al.[8]).

Figure 4. Enclosure of unmeasured state variable x_2 in simultaneous state and parameter estimation for the Lotka-Volterra model.

Figure 5. Enclosure of unmeasured state variable X (biomass concentration) in simultaneous state and parameter estimation for the microbial growth model with Haldane kinetics.

Figure 6. Enclosure of parameters K_S and μ_m in simultaneous state and parameter estimation for the microbial growth model with Haldane kinetics.

Figure 7. Enclosure of unmeasured state variable x_1 (biomass concentration) in simultaneous state and parameter estimation for the three-state biochemical reactor.

Figure 8. Enclosure of parameters k_s and μ_m in simultaneous state and parameter estimation for the three-state biochemical reactor.

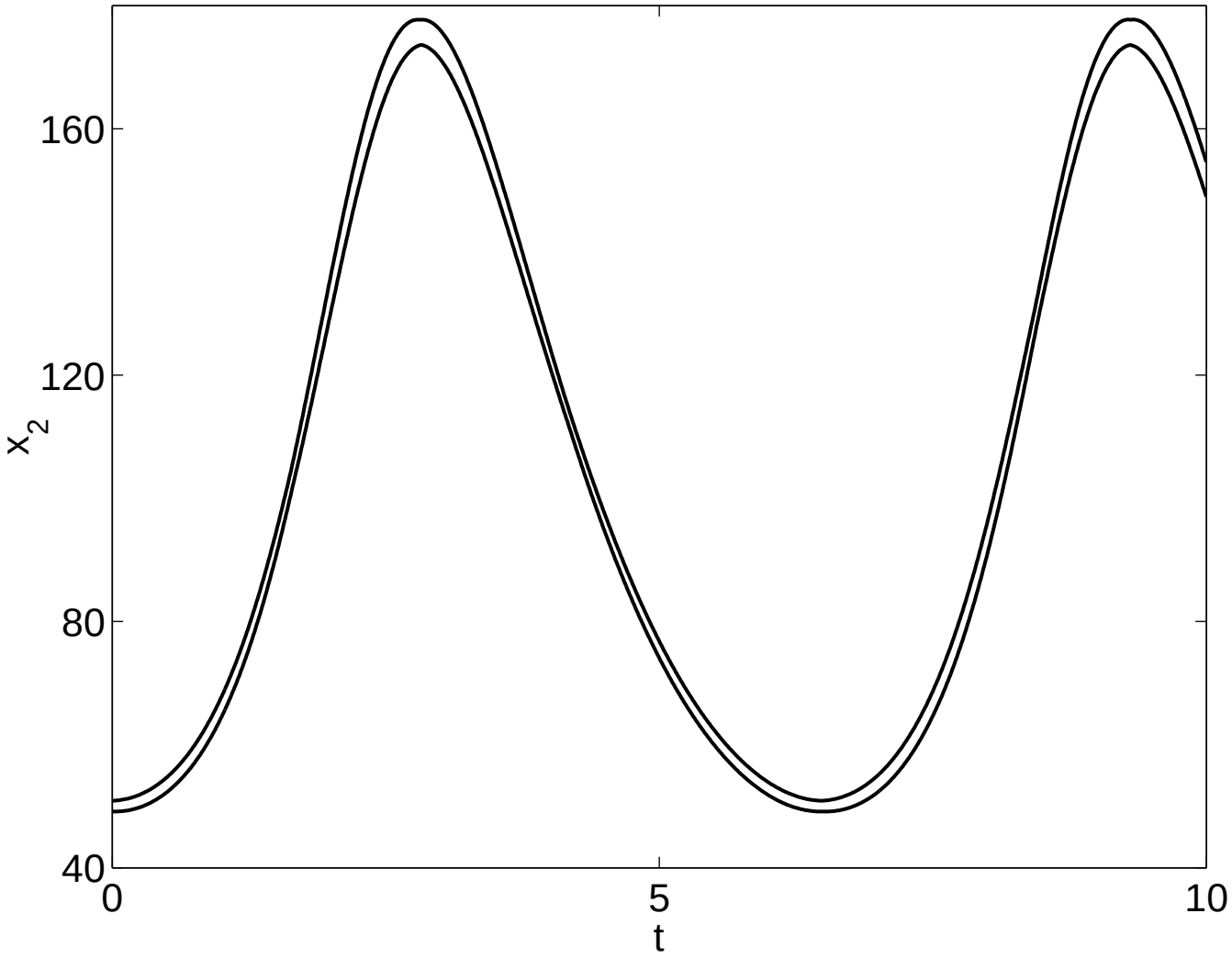

Figure 1: Enclosure of unmeasured state variable x_2 in state estimation for the Lotka-Volterra model.

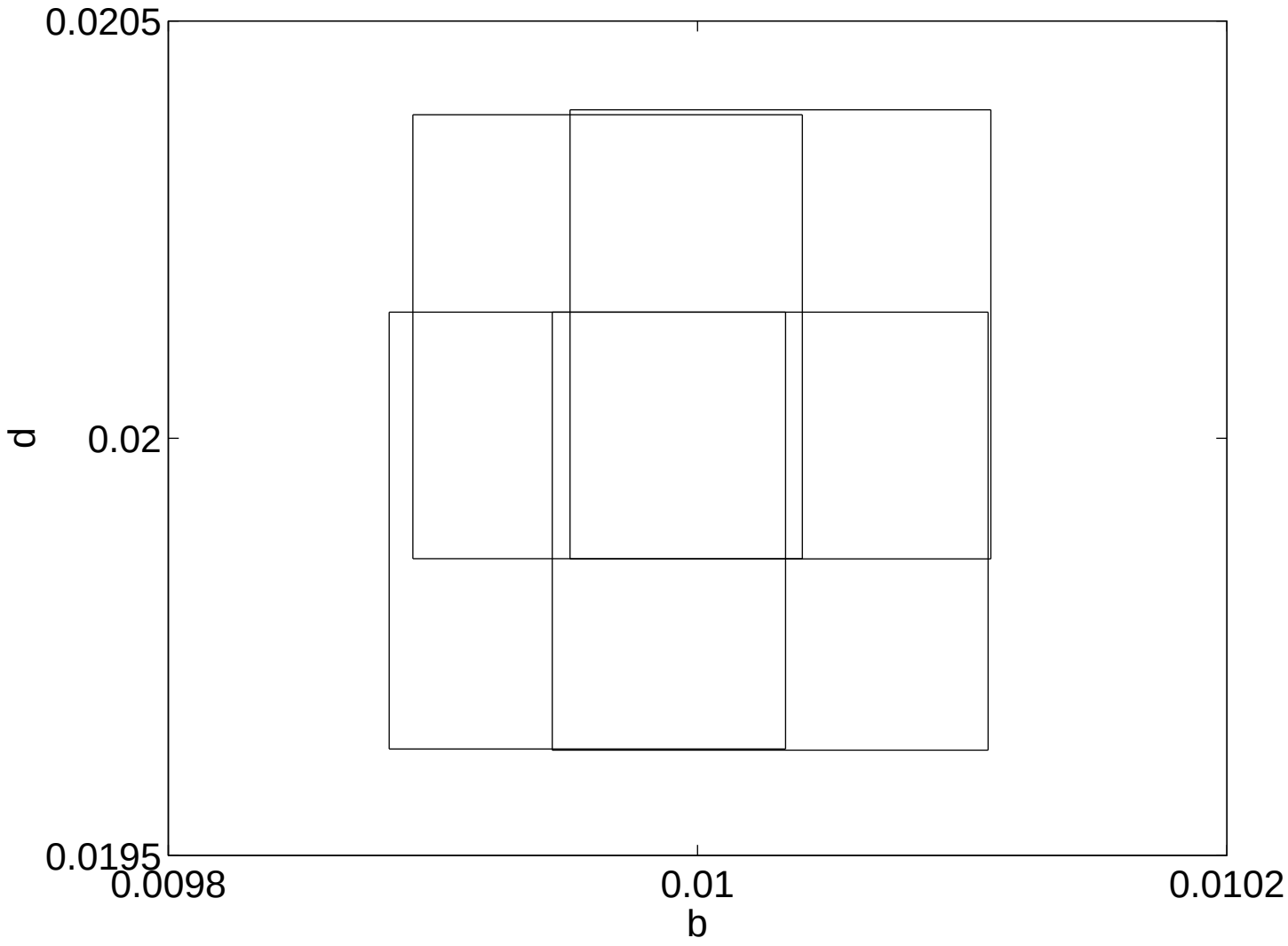

Figure 2: Enclosure of parameters b and d in parameter estimation for the Lotka-Volterra model.

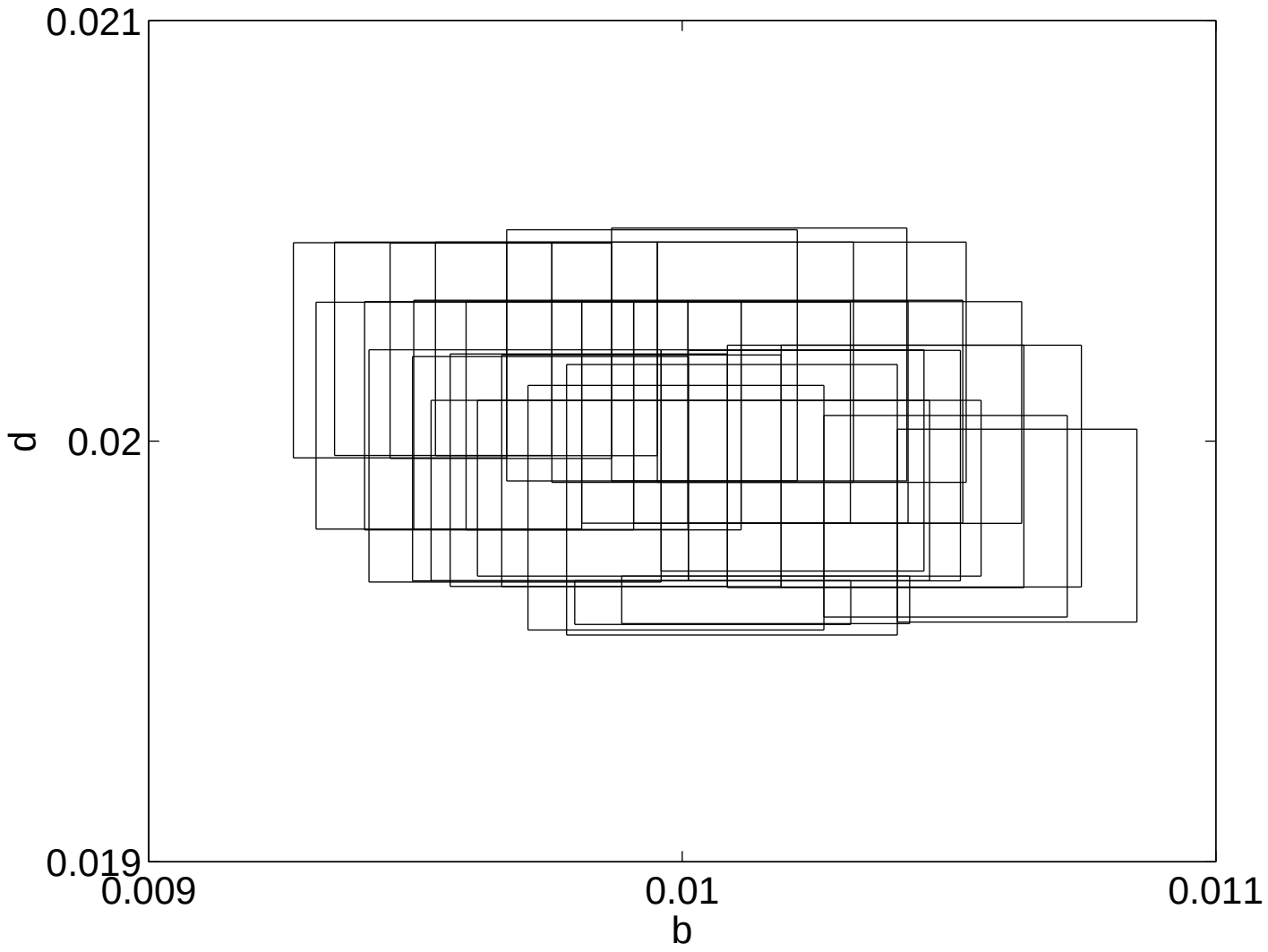

Figure 3: Enclosure of parameters b and d in parameter estimation for the Lotka-Volterra model (using problem data given by Raïssi et al.[8]).

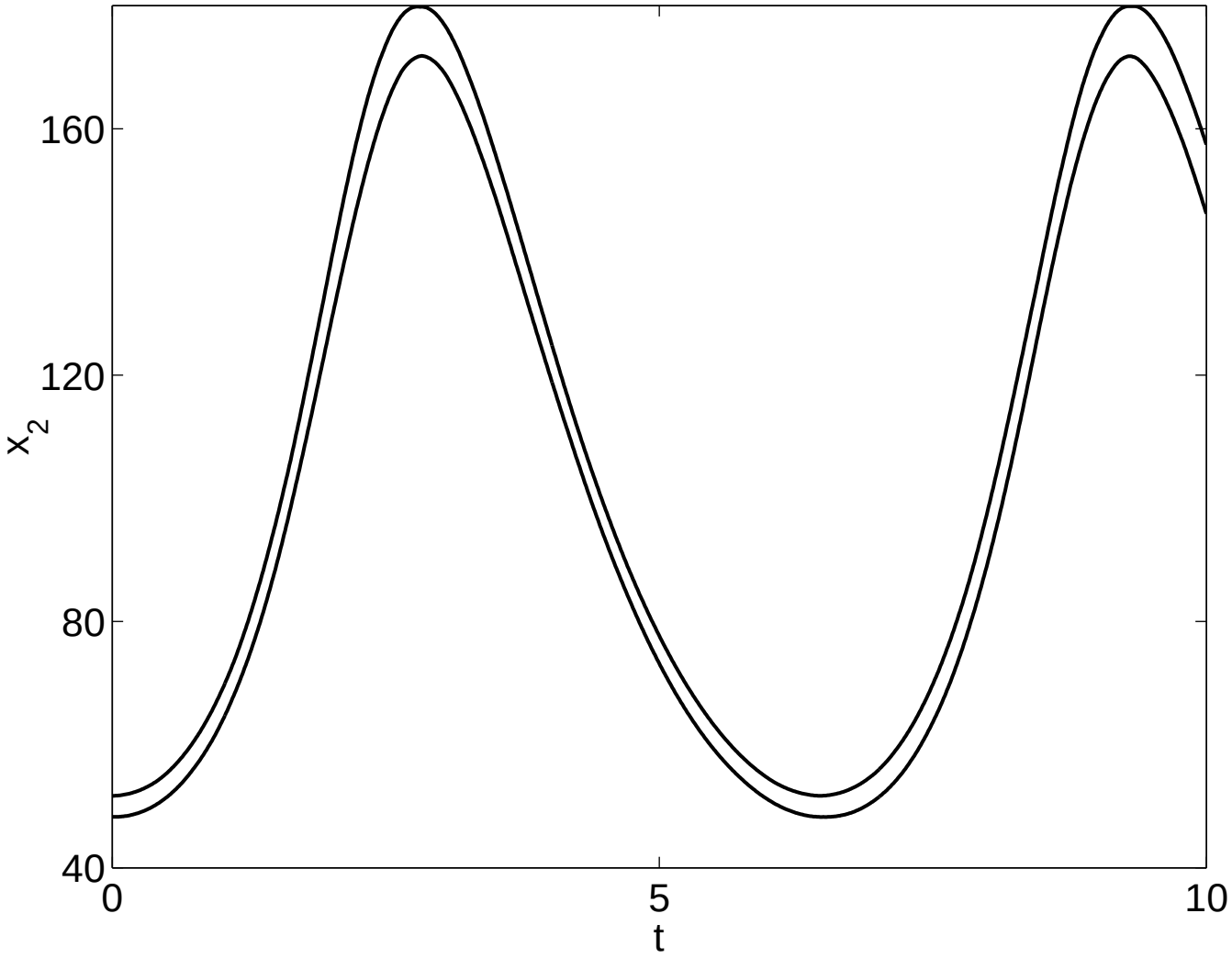

Figure 4: Enclosure of unmeasured state variable x_2 in simultaneous state and parameter estimation for the Lotka-Volterra model.

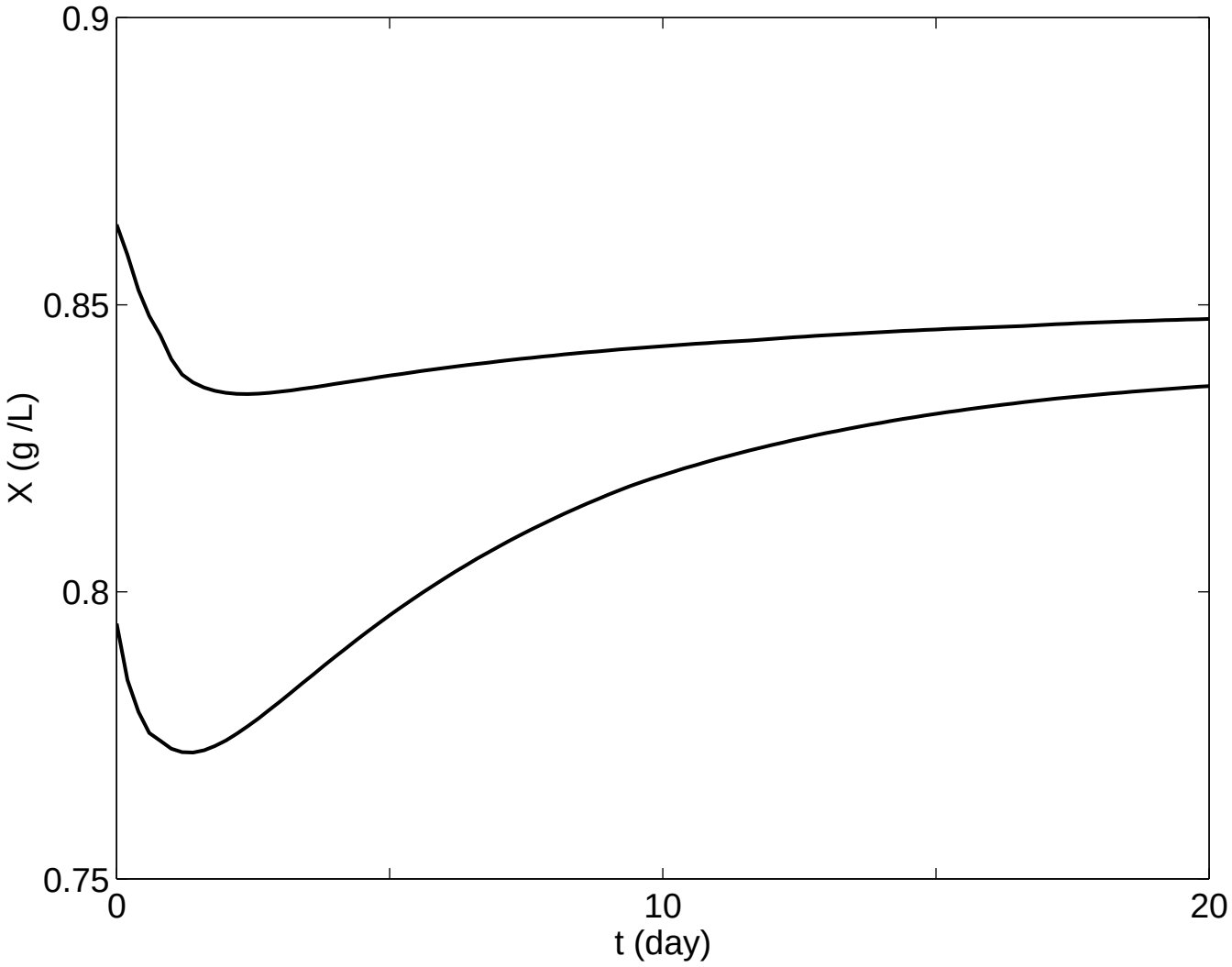

Figure 5: Enclosure of unmeasured state variable X (biomass concentration) in simultaneous state and parameter estimation for the microbial growth model with Haldane kinetics.

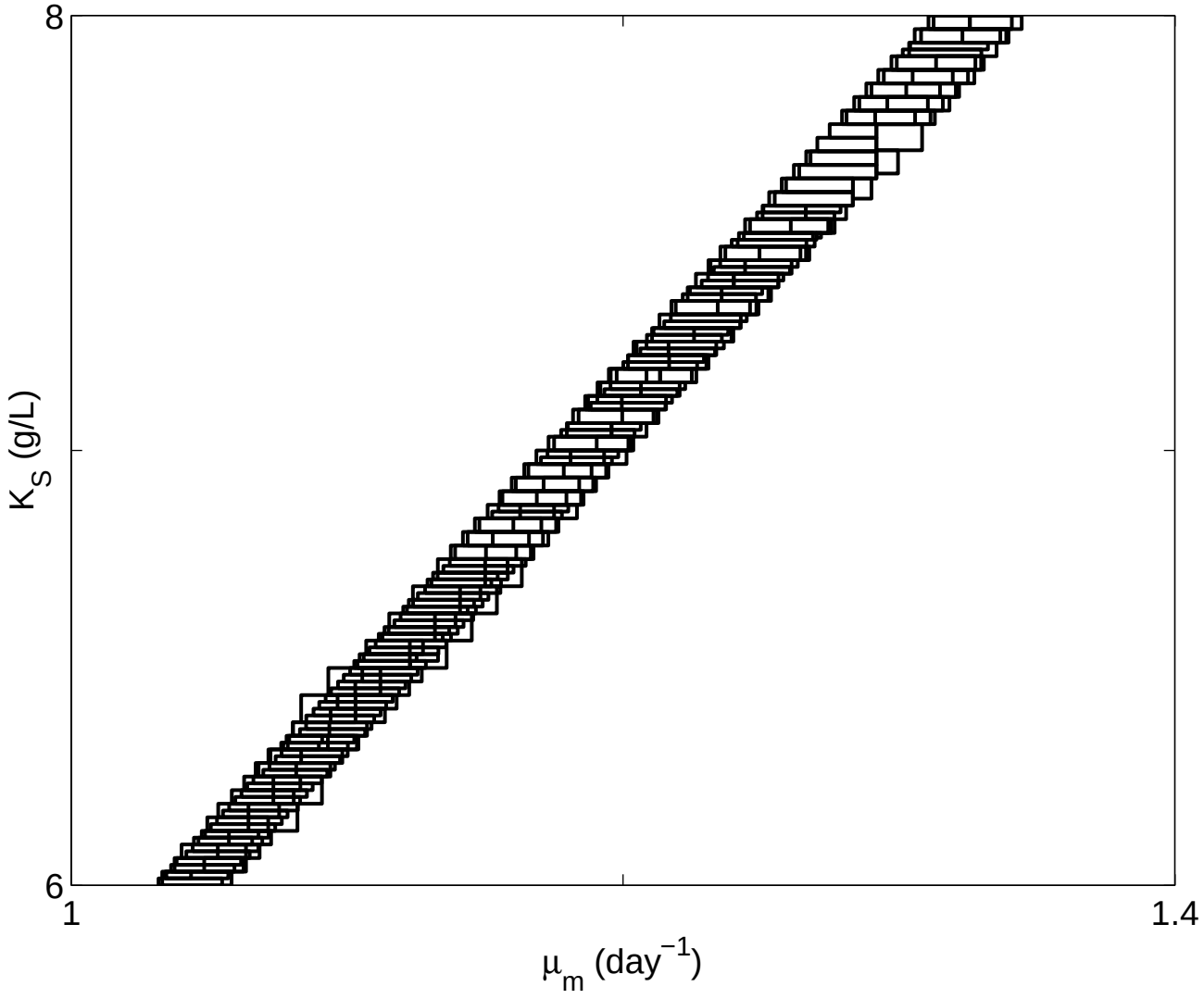

Figure 6: Enclosure of parameters K_S and μ_m in simultaneous state and parameter estimation for the microbial growth model with Haldane kinetics.

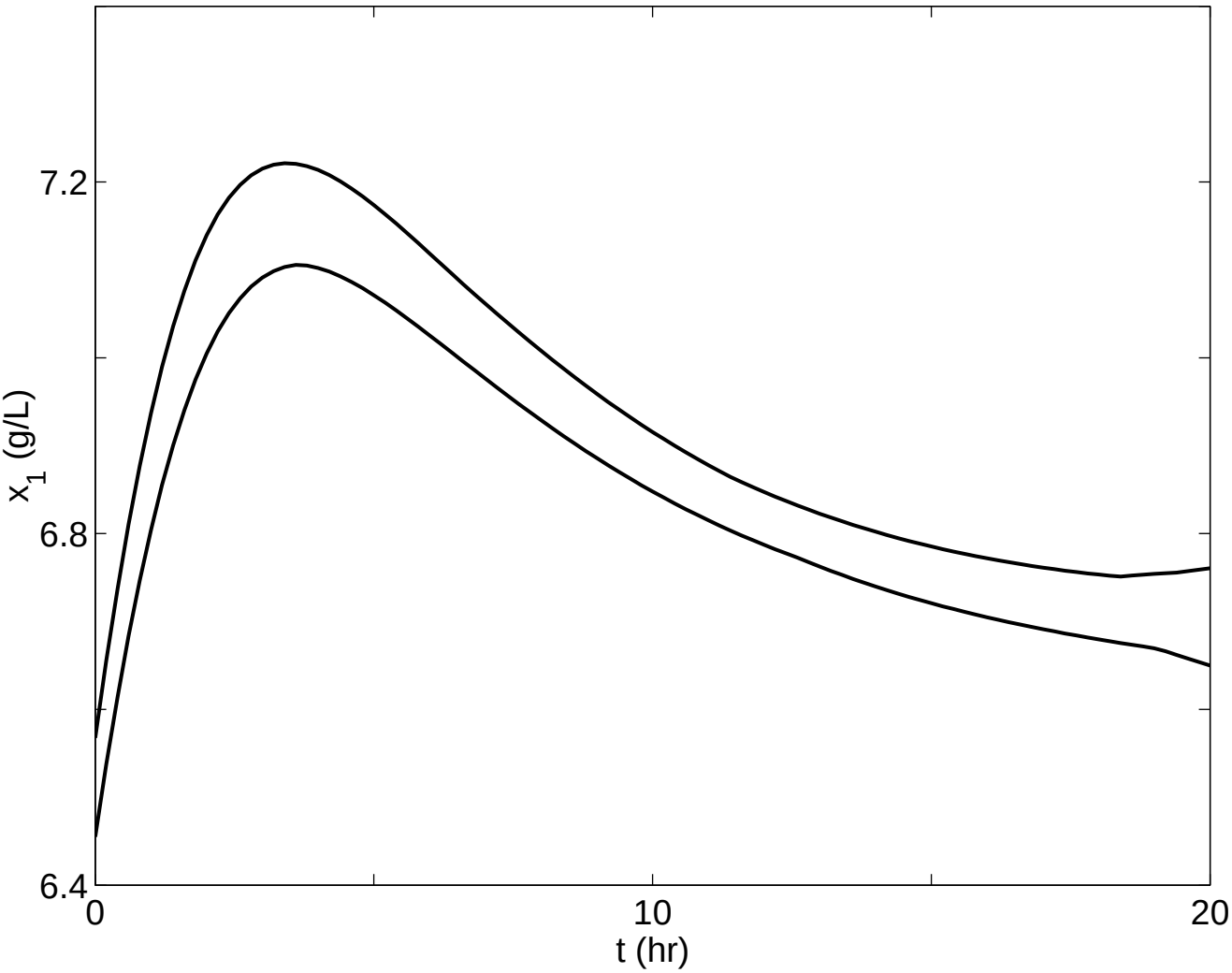

Figure 7: Enclosure of unmeasured state variable x_1 (biomass concentration) in simultaneous state and parameter estimation for the three-state biochemical reactor.

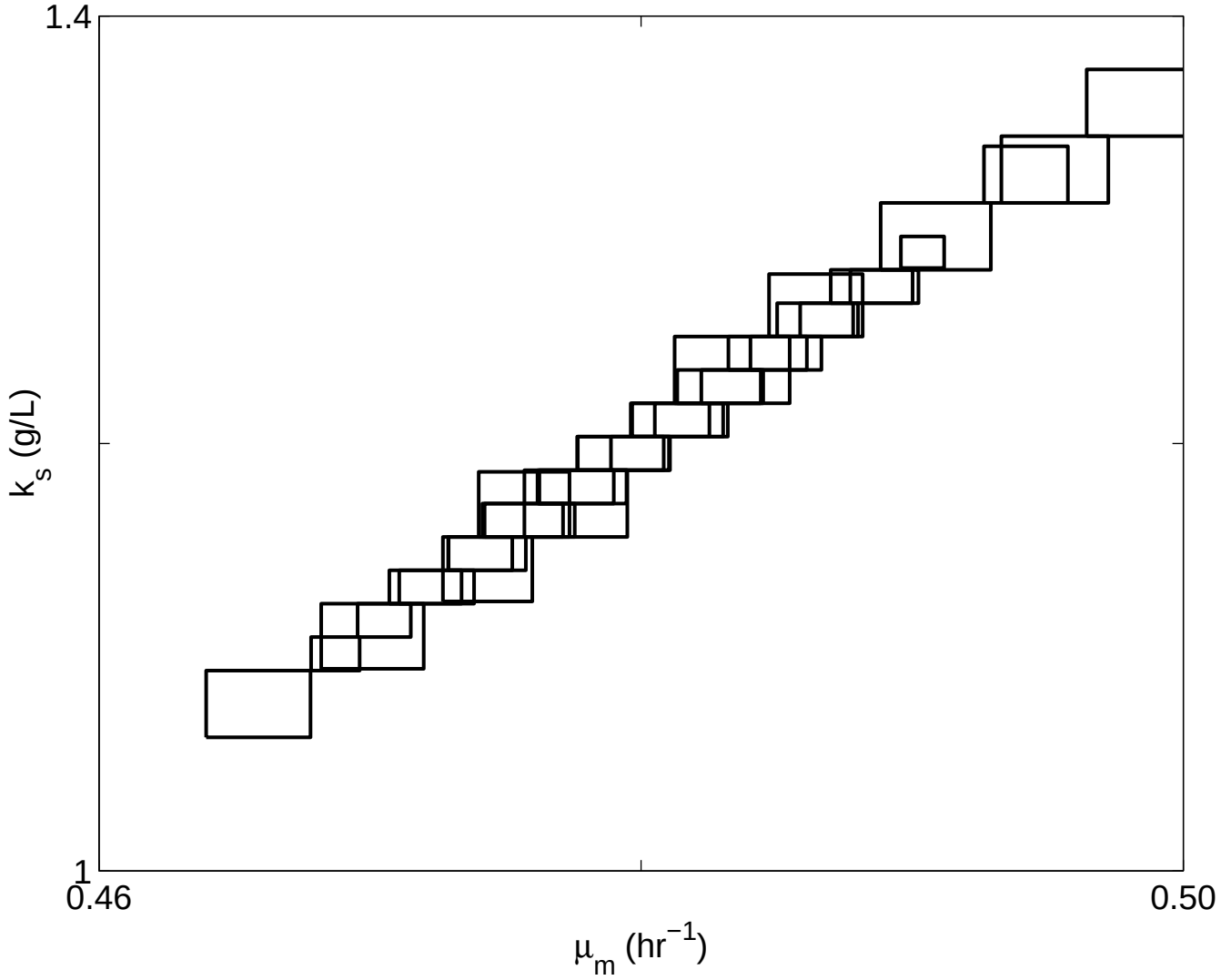

Figure 8: Enclosure of parameters k_s and μ_m in simultaneous state and parameter estimation for the three-state biochemical reactor.