Automatica 44 (2008) 2422–2426

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Brief paper

Rigorous parameter reconstruction for differential equations with noisy data☆

Tomas Johnson [a,*], Warwick Tucker [b]

[a] Department of Mathematics, Uppsala University, Box 480, 751 06 Uppsala, Sweden

[b] Department of Mathematics, University of Bergen, Johannes Brunsgate 12, 5008 Bergen, Norway

ARTICLE INFO

Article history:
Received 20 June 2007
Received in revised form
7 January 2008
Accepted 30 January 2008
Available online 6 May 2008

Keywords:
Rigorous numerics
Parameter estimation
Ordinary differential equations
Interval analysis

ABSTRACT

We present a method that – given a data set, a finitely parametrized system of ordinary differential equations (ODEs), and a search space of parameters – discards portions of the search space that are inconsistent with the model ODE and data. The method is completely rigorous as it is based on validated integration of the vector field. As a consequence, no consistent parameters can be lost during the pruning phase. For data sets with moderate levels of noise, this yields a good reconstruction of the underlying parameters. Several examples are included to illustrate the merits of the method.

1. Introduction

Mathematical models based on differential equations, also called state equations, often depend on one or several parameters that alter the system's behaviour. In many situations, the parameters are unknown and must be estimated using experimental data. These types of *inverse problems* occur in many fields, including pharmacokinetics, systems biology, and ecology. Parameter reconstruction aims at locating parameters that tune the model into a good agreement with the observed data — this ability is important for accurate simulations. To estimate state and parameter variables given a priori bounds on their values is sometimes known as bounded-error estimation (Fogel & Huang, 1982; Schweppe, 1968). The problem of estimating parameters can formally be seen as a special case of state estimation (Jaulin, 2002; Moore, 1992; Raïssi, Ramdani, & Candau, 2005) by simply adding one state variable with a corresponding zero equation for each parameter. This is, however, not the case in practice. Unknown parameters mean that the state equations are not exactly known, so that instead of studying one state equation, an entire set of equations has to be studied. This makes the task of reconstructing or estimating parameters a much harder task than to estimate state variables, given exactly known equations. In fact, during the pruning phase, our method will also estimate the state variables.

Traditionally, the *best-fit* parameters are determined by minimizing a least-square residual error. This recasts the reconstruction to a constrained global optimization problem. Our approach, however, aims at locating the entire *set* of parameters that are *consistent* with the data. This is especially valuable when the data is not exact, but comes with a certain amount of uncertainty. As a side-effect, producing the full set of consistent parameters illustrates the sensitivity of the model equation with respect to data contamination. This kind of information is usually out of reach for classical methods, which must rely on local techniques based on the variational equations.

The underlying mathematics stems from the theory of set-valued computations — *interval analysis*. Although this approach has already been used in the context of parameter reconstruction (Granvilliers, Cruz, & Barahona, 2004; Jaulin, Kieffer, Didrit, & Walter, 2001; Raïssi, Ramdani, & Candau, 2004; Tucker & Moulton, 2006; Tucker, Kutalik, & Moulton, 2007; Walter & Kieffer, 2007) our method is novel in that it handles sparse, partial data sets well, and it does not depend on the decoupling of the ODEs. Decoupling means that each state variable in a system can be handled indivudually, see Tucker and Moulton (2006) and Tucker et al. (2007). Partial data sets are unavoidable in situations when only some of the state variables can be observed. The ability to handle relatively sparse data sets is crucial for the applicability of the method in more realistic situations, when obtaining samples

☆ This paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Wolfgang Scherrer under the direction of Editor Torsten Söederström.

* Corresponding author.
E-mail addresses: johnson@math.uu.se (T. Johnson), warwick.tucker@math.uib.no (W. Tucker).

0005-1098/$ – see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2008.01.032

Table 1
Performance for the two-compartment example

Noise	k_{01} enclosure	k_{12} enclosure	k_{21} enclosure	CPU time
None	[0.4638, 0.5380]	[0.4653, 0.5398]	[0.4989, 0.5014]	00:56:15
1%	[0.3845, 0.6689]	[0.3894, 0.6799]	[0.4870, 0.5249]	01:04:57
2%	[0.3601, 0.7373]	[0.3662, 0.7519]	[0.4785, 0.5456]	02:17:26
5%	[0.3051, 0.9790]	[0.3149, 1.0302]	[0.4516, 0.6323]	02:49:48

is costly or even perhaps involves a human risk (e.g. in drug modeling).

2. Background

Interval analysis is the mathematical foundation of auto-validating algorithms. By computing with intervals instead of single numbers, important properties, such as the continuum of the real line can be captured and used in the algorithms. This leads to very robust methods, well suited for non-linear, global problems. We denote intervals by $\mathbb{X}$, and as soon as one variable in an expression is replaced by an interval we always mean the interval hull of the corresponding expression.

Auto-validating algorithms produce mathematically correct results, incorporating not only the computer's internal representation of the floating point numbers and its rounding procedures, but also all approximation errors of the employed numerical method. Thus the computed result comes equipped with guaranteed error bounds. Areas of success include global optimization, non-linear dynamics, and control theory, see e.g. Alefeld and Herzberger (1983), Jaulin et al. (2001), Moore (1966), Neumaier (1990) and Raïssi et al. (2004). In this paper, we are primarily interested in validated integration of ordinary differential equations — the basics are provided in Section 2.1. A thorough review of these methods is given in Nedialkov, Jackson, and Corliss (1999), and a new method to integrate parametric ODEs, using Taylor models (Makino & Berz, 2003), is given in Lin and Stadtherr (2007).

2.1. Validated integration of ODEs

Most validated ODE solvers are based on a two-stage Taylor series approach, as described in Lohner (1988), Moore (1966) and Nedialkov et al. (1999). We use the solver VNODE-LP written by Nedialkov (0000). The theory behind this solver can be found in Nedialkov and Jackson (1999), Nedialkov et al. (1999) and Nedialkov, RJackson, and Pryce (2001). The first stage of a Taylor series based validated solver proves existence and uniqueness of a solution with a given interval initial value, initial and end times. If successful, the first stage also provides a coarse enclosure of the trajectories. The second stage of a Taylor series method uses this coarse enclosure for the entire piece of trajectory to get a narrow enclosure of the image at the end time point.

We are interested in the set-valued initial value problem

$$\begin{cases} \dot{x} \in f(x; \mathbb{P}) \\ x(t_0) \in \mathbb{X}_0, \end{cases} \tag{1}$$

where $\mathbb{P}$ is a set of model parameters, and $\mathbb{X}_0$ is a set of possible initial conditions. In what follows, all such sets are rectangular boxes. We denote a solution to (1) at time t by $\Phi_{\mathbb{P}}(x(t_0), t - t_0)$, and at a time t_i by $\mathbb{X}_i$, where we assume that $t_{i-1} < t_i$ is some given sequence of times. We remind the reader that $\Phi_{\mathbb{P}}$ denotes the interval hull of the flow Φ_p for $p \in \mathbb{P}$. By inclusion monotonicity, we always have the enclosure

$$x(t_0) \in \mathbb{X}_0 \Rightarrow \Phi_{\mathbb{P}}(x(t_0), t - t_0) \subseteq \Phi_{\mathbb{P}}(\mathbb{X}_0, t - t_0).$$

Table 2
Performance for the predator–prey example

Noise	None	$\pm 10^{-3}$	1%
Tolerance	2^{-6}	2^{-6}	2^{-5}
CPU time	01:29:05	02:49:17	03:44:16
A	[7.87, 8.14]	[7.61, 8.4]	[7.35, 8.93]
B	[3.93, 4.07]	[3.80, 4.2]	[3.67, 4.47]
C	[3.93, 4.07]	[3.80, 4.2]	[3.67, 4.20]
D	[1.91, 1.99]	[1.85, 2.05]	[1.79, 2.18]
E	[0.476, 0.525]	[0.459, 0.542]	[0.426, 0.591]
G	[0.985, 1.018]	[0.984, 1.018]	[0.918, 1.050]
H	[0.183, 0.214]	[0.183, 0.224]	[0.131, 0.263]

Noise	2%	5%
Tolerance	2^{-4}	2^{-3}
CPU time	04:20:19	06:30:33
A	[6.30, 9.45]	[6.30, 12.6]
B	[3.15, 4.73]	[3.15, 6.30]
C	[3.15, 4.73]	[2.10, 5.25]
D	[1.53, 2.31]	[1.02, 2.56]
E	[0.328, 0.722]	[0.131, 1.050]
G	[0.918, 1.050]	[0.787, 1.313]
H	[0.0787, 0.315]	[0.000, 0.420]

Table 3
Comparison with methods using fewer tests, with 5% noise

Method	Size of enclosure	CPU time
Forward–Backward	1.0423	0.6923
Forward	2.1933	0.6923
Backward	2.8950	0.8606

3. Method

Recall that our primary goal is to remove parts of a search domain for the model parameters that are inconsistent with the provided data. A secondary, more challenging, goal is to locate parameters that are consistent with the data. This is harder because the entire data set must be considered when proving consistency. Inconsistency, however, only requires a subset of the data, as we explain in more detail below.

The entire process is based on a set of tests coupled with a bisection scheme. More specifically, when the tests are inconclusive, the parameter box is split along its widest side. Both halves are then subjected to the same battery of tests. This procedure ends when the boxes have reached a smallest tolerable size. The output of this global process is made up of three lists `cList, iList`, and `sList` containing parameter boxes that are consistent, inconsistent, and small, respectively. Geometrically, the contents of `sList` form a set that separates the contents of `cList` and `iList`. As enclosures of the consistent parameters, we take the interval hull of the union of `cList` and `sList`. In general these enclosures consist largely of parameters that have been proved to be inconsistent, since typically the set of consistent parameters is a hypersurface. Thus, more parameters have been excluded by our algorithm than what is indicated in Tables 1–3.

Initially, we attempt to show that the current parameter box $\mathbb{P}$ is consistent with the data. The criterion we use for this is that *some* initial point $x_0 \in \mathbb{X}_0$ should flow through each data range:

$$\exists x_0 \in \mathbb{X}_0 \text{ s.t. } \forall i \, \Phi_{\mathbb{P}}(x_0, t_i - t_0) \subseteq \mathbb{X}_i \Rightarrow \mathbb{P} \text{ is } \textit{consistent} \text{ with the data set.} \tag{2}$$

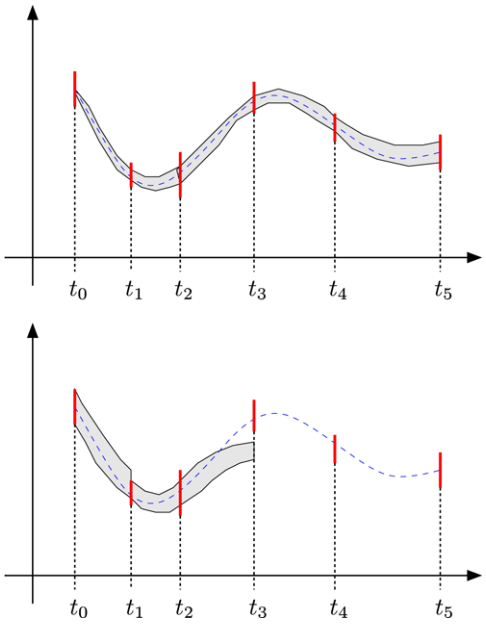

Fig. 1. (a) A consistent parameter; (b) Forward inconsistency proved between t_2 and t_3.

In our implementation, we use the midpoint of $\mathbb{X}_0$ as initial point, see Fig. 1(a). Other choices of x_0 would give other sets of consistent parameters. By taking a set of initial points with some distribution, more parameters would potentially be regarded as consistent. This would, however, decrease the run-time efficiency of the program since the trajectory of each starting point would need to be integrated individually. As is indicated by the fact that 95% of the remaining parameters were proved to be consistent in the example in Section 5, choosing one initial point, the midpoint, seems to be a good compromise. All consistent parameter boxes are removed from further study, and stored in the list `cList`.

If we do not succeed in proving consistency, we switch tactics, and attempt to establish inconsistency. This is done in three stages. We begin by computing the flow of each data range one time-step forward. This is done in an increasing order of time, allowing us to intersect the image of $\mathbb{X}_i$ with $\mathbb{X}_{i+1}$ before flowing the latter:

$$\mathbb{W}_0^+ = \mathbb{X}_0,$$
$$\mathbb{W}_{i+1}^+ = \mathbb{X}_{i+1} \cap \Phi_{\mathbb{P}}(\mathbb{W}_i^+, t_{i+1} - t_i) \quad (i = 0, \ldots, N-1).$$

If any of the intersections are empty — (see Fig. 1(b)), — the corresponding parameter box is removed:

$$\mathbb{W}_{i+1}^+ = \emptyset \Rightarrow \mathbb{P} \text{ is } \textit{forward inconsistent} \text{ between } t_i \text{ and } t_{i+1}. \tag{3}$$

If no inconsistencies are detected during this forward sweep, we flow backwards and use the (possibly tighter) data set $\{t_i, \mathbb{W}_i^+\}_{i=0}^N$ as constraints:

$$\mathbb{W}_N^- = \mathbb{W}_N^+,$$
$$\mathbb{W}_i^- = \mathbb{W}_i^+ \cap \Phi_{\mathbb{P}}(\mathbb{W}_{i+1}^-, t_i - t_{i+1}) \quad (i = N-1, \ldots, 0).$$

Again, if any of the intersections are empty, the corresponding parameter box is removed:

$$\mathbb{W}_i^- = \emptyset \Rightarrow \mathbb{P} \text{ is } \textit{backward inconsistent} \text{ between } t_i \text{ and } t_{i+1}. \tag{4}$$

The forward and backward sweeps reduce the given data to a subset consistent with the parameters. That is, we get state estimation as a side effect of our parameter estimation scheme. The parameter domain is not contracted by the forward and backward sweeps, since parameters are static, its size is only reduced by removing entire boxes.

If no inconsistencies have been revealed during the backward sweep, one final test is performed. This test flows halfway between two consecutive data ranges:

$$\Phi_{\mathbb{P}}\left(\mathbb{W}_i^-, \frac{t_{i+1} - t_i}{2}\right) \bigcap \Phi_{\mathbb{P}}\left(\mathbb{W}_{i+1}^-, \frac{t_i - t_{i+1}}{2}\right) = \mathbb{W}_{i+1/2}.$$

If any of the intersections are empty, the corresponding parameter box is removed:

$$\mathbb{W}_{i+1/2} = \emptyset$$
$$\Rightarrow \mathbb{P} \text{ is } \textit{midway inconsistent} \text{ between } t_i \text{ and } t_{i+1}. \tag{5}$$

This method is very general and can be used for any finitely parametrized differential equation and data set. Naturally, the performance deteriorates with an increasing number of parameters and/or with a sparse data set. In such situations, monotonicity properties of the model ODEs can be of great help. This scenario is studied in Section 5.

4. Direct use

We start by providing two examples in which we apply the algorithm without explicitly using any qualitative information about the differential equations. First, we consider a two-compartment model previously studied in Walter and Kieffer (2007). Although qualitative information is available for this model, we want to illustrate that for simple problems, it is possible to get good results without any a priori information about the system. Our second example, included to show the ability of our method to handle systems with a larger number of parameters, is a predator–prey model. This three-dimensional model with seven parameters has previously been studied in the same context in Willms (2007).

All computations were performed on an Intel Xeon 3.2 Ghz processor with 3072 Mb of RAM. VNODE-LP was used with PROFIL/BIAS (PROFIL/BIAS, 0000) and compiled using GNU C++ 3.4.6 in Linux.

4.1. A two-compartment model

Compartment models are common in e.g. biology, chemical engineering, and pharmacokinetics. They model compartments interacting with each other and the outside. That is, material flows into the system and out of it. In addition, the compartments interact with each other. In what follows, we consider a two-compartment model, previously studied in Walter and Kieffer (2007).

In Walter and Kieffer (2007), the fact that the model is *cooperative* is used. This means that it is possible to enclose the solution between solutions of two extremal vector fields. On the other hand, we use this system as an example of a direct approach and solve the full set-valued system. In fact, apart from the initial state $(x_1(t_0), x_2(t_0))$, we only use measurements of the second component x_2. All we need to know about the unobserved component x_1 is that it does not exceed its initial value: $x_1(t) \in [0, x_1(t_0)]$. The ability to deal with these types of partial data sets is a great advantage of the presented method, compared to those described in Tucker and Moulton (2006) and Tucker et al. (2007) where measurements of all variables are required.

The model ODE is given by:

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} -(k_{21} + k_{01})x_1 + k_{12}x_2 \\ k_{21}x_1 - k_{12}x_2 \end{pmatrix}, \tag{6}$$

where $p = (k_{01}, k_{12}, k_{21})$. The first subscript is the label of the container that is flowed to, and the second number is the container

flowed from. Index 0 represents the outside. The data set was produced using $p^\star = (0.5, 0.5, 0.5)$, and the search domain was chosen as $[0, 5]^3$. The samples were taken at 17 uniformly spaced times from $t = 0$ to $t = 16$ with 0, 1, 2, and 5% relative noise added to the exact data. The results of the reconstruction are shown in Table 1.

It is interesting to note that, despite the wide enclosures for 5% noise, the center of mass of the retained parameter boxes is (0.520, 0.545, 0.523). For the noise-free case, the center of mass is (0.500, 0.501, 0.500). This is a very good *best-fit* candidate. No parameters were proved to be consistent.

4.2. A predator–prey model

The previous example has a small number of parameters, contrary to many systems that occur in applications. To investigate if our method is useful in a more realistic situation, we test it on a three dimensional predator-prey system with seven parameters:

$$\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{pmatrix} = \begin{pmatrix}Ax(1 - H(x + y)) - Bxz\\ Cy(1 - H(x + y)) - Dyz\\ E(2Bxz + Dyz) - Gz\end{pmatrix}. \tag{7}$$

Following Willms (2007), we use the initial condition $(x_0, y_0, z_0) = (3, 1, 2)$ together with the target parameters $A^\star = 8$, $B^\star = 4$, $C^\star = 4$, $D^\star = 1.95$, $E^\star = 0.5$, $G^\star = 1$, and $H^\star = 0.2$ when generating the clean data set. The perturbed data sets were generated with *absolute* noise radius 10^{-3}, to compare with the results in Willms (2007), as well as *relative* noise levels of 1%, 2%, and 5%, respectively.

In the reconstruction, we use a subset of the data set from Willms (2007), namely 201 uniformly distributed samples in $0 \le t \le 10$. The search region for each parameter is $[0, 2.1 \times$ (target parameter)], which corresponds to the maximal search region considered in Willms (2007). The results are shown in the Table 2.

Again, despite the wide enclosures for 5% noise, the center of mass of the retained parameter boxes (8.239, 4.187, 4.097, 2.007, 0.5673, 1.021, 0.207) is a decent *best-fit* candidate. No parameters were proved to be consistent.

4.2.1. The effect of the inconsistency tests

To compare the impact the various tests, as described in Section 3, have on the size of the enclosure, and the run-time of the algorithm, we use the 5% noise case of the predator–prey model. The results are shown in Table 3, where all numbers are normalized with respect to the full version of the algorithm.

5. Monotonicity

To improve accuracy of the algorithm, qualitative information about the flow can be taken into account. We are primarily interested in monotonicity, which allows us to flow end points instead of intervals. We study the SIR system, which is a simple model of the spread of an infectious disease. This system has few variables and few parameters, which allows us to get very narrow results. In fact, for the experimental data considered, taken from Granvilliers et al. (2004), we prove that 95% of the remaining feasible parameters are consistent with the data. The low total number of dimensions involved, four, allows us to make direct use of monotonicity without any further analysis and simply flow the 16 corners of each box and take the hull of the result.

5.1. The SIR model

A simple model to describe the spread of an infectious disease is SIR. It divides the population into three disjoint groups, S (susceptibles), I (infectives) and R (recovered). A standing assumption is that once an individual has recovered, she cannot be reinfected. Furthermore, the total population size is assumed to be constant, that is, $(S + I + R)' = 0$. The system of ODEs is

$$\begin{pmatrix}\dot{S}\\ \dot{I}\\ \dot{R}\end{pmatrix} = \begin{pmatrix}-rSI\\ rSI - aI\\ aI\end{pmatrix}, \tag{8}$$

where r is the disease transmition frequency, and a is the recovery rate. Using that the population size is constant allows us to reduce the phase space by one dimension. In particular, we can remove the variable R, producing the reduced system

$$\begin{pmatrix}\dot{S}\\ \dot{I}\end{pmatrix} = \begin{pmatrix}-rSI\\ rSI - aI\end{pmatrix}. \tag{9}$$

Rather than integrating with VNODE directly, we observe the following properties of the vector field:

$$\begin{aligned}&\frac{\partial \dot{S}}{\partial S} = -rI \quad \frac{\partial \dot{S}}{\partial I} = -rS \quad \frac{\partial \dot{S}}{\partial r} = -IS\\ &\frac{\partial \dot{I}}{\partial S} = rI \quad \frac{\partial \dot{I}}{\partial I} = rS - a \quad \frac{\partial \dot{I}}{\partial r} = SI \quad \frac{\partial \dot{I}}{\partial a} = -I.\end{aligned} \tag{10}$$

The only non-monotonic relationship is $\frac{\partial \dot{I}}{\partial I}$, which is zero if $rS - a = 0$. A local extrema of $\dot{I}$ therefore must occur when S, r, and a are at endpoints. The flow in the SI-phase plane of an I-interval is the flow of a line of initial conditions for a smooth two-dimensional autonomous vector field, so solutions cannot cross each other in the phase plane. Indeed, we have

$$\frac{\dot{I}}{\dot{S}} = \frac{rIS - aI}{-rIS} = -1 + \frac{a}{rS},$$

so the solution curves are

$$I(S) = -S + \frac{a}{r}\ln(S) + C \quad (C \in \mathbb{R}).$$

Therefore a solution either decreases monotonically in I or first increases monotonically and then decreases monotonically, since S decreases monotonically. Thus, the largest I value is the result of flowing the largest I value in a box, and the smallest I value is the result of flowing the smallest I value in a box. Hence, it suffices to flow the 16 corners of a (S, I, r, a)-box and take the hull of the results. This procedure increases the speed and accuracy of the algorithm tremendously. We run the program with the following real-life data, also used for the same problem in Granvilliers et al. (2004). It describes an influenza epidemic in an English Boarding school initiated by a single boy from a population of 763. Allowing for a ± 30 absolute error in the measured data yields the data set presented in Table 4.

The search domain used was $r \in [0, 0.01]$ and $a \in [0, 1]$; the norm in the parameter domain was scaled with a factor of 100 in r so that both parameters were split an equal number of times. Therefore, the weighted initial domain has unit volume. In Table 5, we give two different measures of the retained parameters. The column "remaining volume" lists the volume of all small and consistent boxes; the column "consistent volume" lists the volume of all consistent boxes.

Using $\texttt{tol} = 2^{-14}$, the enclosures of r and a are

$$r \in [2.14, 2.22] \times 10^{-3} \quad a \in [4.25, 4.66] \times 10^{-1},$$

and their centres of mass are 2.18×10^{-3} and 4.45×10^{-1}, in good agreement with the results presented in Granvilliers et al. (2004). Our method, however, also proves that 95% of the retained set of parameters are consistent with the data set.

Table 4
The number of infected/susceptible patients

t	I_{measured}	$I_{\text{initial value}}$	$S_{\text{initial value}}$
0	1	[1, 1]	[762, 762]
3	22	[0, 52]	[0, 762]
4	78	[48, 108]	[0, 715]
5	222	[192, 252]	[0, 571]
6	300	[270, 330]	[0, 493]
7	256	[226, 286]	[0, 493]
8	233	[203, 263]	[0, 493]
9	189	[159, 219]	[0, 493]
10	128	[98, 158]	[0, 493]
11	72	[42, 102]	[0, 493]
12	28	[0, 58]	[0, 493]
13	11	[0, 41]	[0, 493]
14	6	[0, 36]	[0, 493]

Table 5
A summary of the SIR example

Tolerance	Remaining volume	Consistent volume	Run time
2^{-10}	2.26736×10^{-4}	6.306×10^{-5}	00:30:30
2^{-12}	1.66655×10^{-4}	1.316×10^{-4}	03:04:56
2^{-14}	1.54317×10^{-4}	1.459×10^{-4}	08:32:07

6. Conclusions

We have presented a method to estimate parameters from noisy data, continuing in the spirit of Tucker and Moulton (2006) and Tucker et al. (2007). We do not focus on finding one best-fit parameter by solving a global optimization problem, for instance formulated as a least squares approximation of the data points. Instead, we focus on consistency by discarding those portions of the parameter domain that are inconsistent with the given data. Equally important, we are able to find parameters that are provably consistent with the data. Our method allows for noise in the data (which may be partial), and we give several examples on how the set of inconsistent parameters recedes as the noise level increases. Our interpretation of noise and noise levels is that the user of our algorithm has a priori knowledge of the amount of noise in the data, e.g. measurement errors. Therefore, since data is normally given as points at each measured time, the data has to be pre-processed into intervals using suitable absolute or relative noise levels.

We demonstrate how to prove and implement qualitative properties of the flow, primarily monotonicity, to improve speed and accuracy of the flowing algorithm. The drawback of this approach, as we have implemented it, is that it requires information about the specific problem at hand. Further research is needed to automate monotonicity checks using automatic differentiation techniques.

References

Alefeld, G., & Herzberger, J. (1983). *Introduction to interval computations*. New York: Academic Press.

Fogel, E., & Huang, Y. F. (1982). On the value of information in system identification—bounded noise case. *Automatica–Journal of IFAC*, *18*(2), 229–238.

Granvilliers, L., Cruz, J., & Barahona, P. (2004). Parameter estimation using interval computations. *SIAM Journal of Scientific Computing*, *26*(2), 591–612.

Jaulin, L. (2002). Nonlinear bounded-error state estimation of continuous-time systems. *Automatica Journal of IFAC*, *38*(6), 1079–1082.

Jaulin, L., Kieffer, M., Didrit, O., & Walter, E. (2001). *Applied interval analysis*. Springer-Verlag.

Lin, Y., & Stadtherr, M. A. (2007). Validated solutions of initial value problems for parametric ODEs. *Applied Numerical Mathematics*, *57*(10), 1145–1162.

Lohner, R. (1988). Einschließung der Lösung gewöhnlicher Anfangs- und Randwertaufgaben und Anwendungen. Ph.D. thesis. Universität Karlsruhe.

Makino, K., & Berz, M. (2003). Taylor models and other validated functional inclusion methods. *International Journal of Pure and Applied Mathematics*, *6*(3), 239–316.

Moore, R. E. (1966). *Interval analysis*. Englewood Cliffs, New Jersey: Prentice-Hall.

Moore, R. E. (1992). Parameter sets for bounded-error data. *Mathematics of Computation and Simulation*, *34*, 113–119.

Nedialkov, N. S., & Jackson, K. R. (1999). An interval Hermite–Obreschkoff method for computing rigorous bounds on the solution of an initial value problem for an ordinary differential equation. *Reliable Computing*, *5*(3), 289–310.

Nedialkov, N. S., Jackson, K. R., & Corliss, G. F. (1999). Validated solutions of initial value problems for ordinary differential equations. *Applied Mathematics and Computation*, *105*(1), 21–68.

Nedialkov, N. S., RJackson, K., & Pryce, J. D. (2001). An effective high-order interval method for validating existence and uniqueness of the solution of an IVP for an ODE. *Reliable Computing*, *7*(6), 449–465.

Nedialkov, N. S. VNODE-LP a validated solver for initial value problems in ordinary differential equations, available at www.cas.mcmaster.ca/nedialk/Software/VNODE/VNODE.shtml.

Neumaier, A. (1990). *Encyclopedia of mathematics and its applications*: *Vol. 37*. *Interval methods for systems of equations*. Cambridge: Cambridge Univ. Press.

PROFIL/BIAS, Available at www.ti3.tu-harburg.de/Software/PROFILEnglisch.html.

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

Raïssi, T., Ramdani, N., & Candau, Y. (2005). Bounded error moving horizon state estimator for non-linear continuous-time systems: Application to a bioprocess system. *Journal of Process Control*, *15*, 537–545.

Schweppe, F. C. (1968). Recursive state estimation: Unknown but bounded errors and system inputs. *IEEE Transactions on Automatic Control*, *13*(1), 22–28.

Tucker, W., & Moulton, V. (2006). Parameter reconstruction for biochemical networks using interval analysis. *Reliable Computing*, *12*(5), 389–402.

Tucker, W., Kutalik, Z., & Moulton, V. (2007). Estimating parameters for generalized mass action models using constraint propagation. *Mathematical Biosciences*, *208*, 607–620.

Willms, A. R. (2007). Parameter range reduction for ODE models using cumulative backward differentiation formulas. *Journal of Computational and Applied Mathematics*, *203*(1), 87–102.

Walter, E., & Kieffer, M. (2007). Guaranteed nonlinear parameter estimation in knowledge-based models. *Journal of Computational and Applied Mathematics*, *199*(2), 277–285.

Tomas Johnson was born in Örebro, Sweden, in 1979. He received his M.Sc. in Engineering Physics from Uppsala University in 2005, and is since then persuing a Ph.D. within the CAPA group at Bergen University. His research interests include parameter estimation, global optimisation, and computational aspects of nonlinear dynamical systems.

Warwick Tucker was born in Sydney, Australia, in 1970. He received his Ph.D. in mathematics at Uppsala University in 1998, and is currently leading the CAPA group at Bergen University. His research field is Computer-Aided Proofs in Analysis, with applications such as nonlinear dynamical systems, parameter estimation, and spectral theory.