Interval constraint propagation with application to bounded-error estimation

Luc Jaulin

Interval constraint propagation with application to bounded-error estimation

Luc Jaulin

Laboratoire des Signaux et Systèmes,
CNRS, Supélec, Plateau de Moulon,
91192 Gif-sur-Yvette Cedex, France.

On leave from
Laboratoire d'Ingénierie des Systèmes Automatisés,
ISTIA, 62 avenue Notre Dame du Lac,
49000 Angers, France

Phone: (33) 1 69 85 17 21
Fax: (33) 1 69 41 30 60
Email: jaulin@lss.supelec.fr

Suggested running title: Consistency Techniques for Estimation.

Abstract: For a large class of bounded-error estimation problems, the posterior feasible set $\mathcal{S}$ for the parameters can be defined by nonlinear inequalities. The set-inversion approach combines classical interval analysis with branch-and-bound algorithms to characterize $\mathcal{S}$. Unfortunately, as bisections have to be done in all directions of the parameter space, this approach is limited to problems involving a small number of parameters. Techniques based on interval constraint propagation make it possible to drastically reduce the number of bisections. In this paper, these techniques are combined with set inversion to bracket $\mathcal{S}$ between inner and outer subpavings (union of nonoverlapping boxes). When only interested in the feasible intervals for the parameters, the set inversion approach becomes inefficient, and a new algorithm able to compute these intervals is given. This algorithm uses a new interval-based local research to compute the smallest box that contains $\mathcal{S}$. It is then compared with existing methods on an example taken from the literature.

Keywords: Bounded-error estimation, Constraint propagation, Interval analysis, Nonlinear constraint solving, Set inversion.

1 Introduction

This paper is concerned with estimating the unknown parameters of a model from experimental data collected on a system, in a bounded-error context (see, *e.g.* Walter, 1990; Norton, 1994; Norton, 1995; Milanese *et al.*, 1996; Walter and Pronzato, 1997). Denote by $\vec{y}$ the vector of all these data and by $\vec{p}$ the parameter vector to be estimated. Assume that a parametric model structure $\mathcal{M}(.)$ (*i.e.* a set of models parametrized by $\vec{p}$) is available for the system. For the experimental conditions used, each model $\mathcal{M}(\vec{p})$ generates a vector output $\vec{f}(\vec{p})$ homogeneous to the data vector $\vec{y}$. If $[\vec{y}]$ denotes the axis-aligned box of all admissible output vectors, and $[\vec{p}]$, the *prior* feasible axis-aligned box for the parameters, *the posterior feasible set* is defined by

$$\mathcal{S} = \vec{f}^{-1}([\vec{y}]) \cap [\vec{p}] \tag{1}$$

where $\vec{f}^{-1}(.)$ is the reciprocal image of $\vec{f}$ in a set theoretical sense. The problem to be solved, known as *set inversion problem*, is that of characterizing $\mathcal{S}$ in a guaranteed way. When the vector function $\vec{f}$ is affine in $\vec{p}$, efficient and accurate methods exist to enclose $\mathcal{S}$ in an ellipsoid or a box (Belforte *et al.*, 1990; Fogel and Huang, 1982; Milanese and Belforte, 1982) or to characterize $\mathcal{S}$ exactly (*e.g.,* Walter and Piet-Lahanier, 1989). When $\vec{f}$ is nonlinear, simple-shaped sets that contain $\mathcal{S}$ can be computed sometimes (*e.g.* Norton, 1987; Milanese and Vicino, 1991). In this paper, the case where $\vec{f}$ is nonlinear is considered and two kinds of characterization for $\mathcal{S}$ are of interest. The first one consists in bracketing $\mathcal{S}$ between inner and outer subpavings (a *subpaving* is an union of nonoverlapping boxes) with an arbitrary precision. The second one consists in finding the smallest axis-aligned box (also called interval hull) $[\mathcal{S}]$ which contains $\mathcal{S}$.

When the number of parameters is small, the algorithm SIVIA (Set Inversion Via Interval Analysis) presented in (Jaulin and Walter, 1993) combines branch-and-bound techniques with interval analysis (Moore, 1979) to bracket $\mathcal{S}$ between two subpavings with an arbitrary precision. But, as bisections have to be performed in all directions of the parameter space, SIVIA is limited to problems involving only few parameters. In this paper, SIVIA is used with *interval constraint propagation (*ICP) techniques in order to reduce the number of bisections to be performed. Interval constraint propagation (also called *local consistency* approach in the literature), was pioneered by Cleary (1987) and Davis (1987). It combines constraint propagation techniques, classically used in the domains of artificial intelligence (Mackworth, 1977), with interval analysis. Constraint propagation techniques were introduced by Waltz (1975) to address combinatorial problems over finite sets and have been intensively studied since then.

When only interested in the feasible intervals for the parameters, many computations performed by SIVIA become useless and a new efficient algorithm able to compute these intervals is given. This algorithm alternates a new interval-based local research with a elimination procedure to bracket the interval hull $[\mathcal{S}]$ of $\mathcal{S}$. The components of $[\mathcal{S}]$ are the feasible intervals for the parameters.

The paper is organized as follows. ICP techniques are briefly presented in section 2. In Section

3, the new version of the algorithm SIVIA, based on ICP, is presented and compared with the former SIVIA with a test-case taken from (Milanese and Vicino, 1991). In Section 4, the algorithm which computes the interval hull of $\mathcal{S}$ is presented and a comparison with the results obtained by Milanese and Vicino (1991) is shown. A notation table is given on page 14.

2 INTERVAL CONSTRAINT PROPAGATION

Interval constraint propagation (ICP) makes it possible (Benhamou and Older, 1997) to generate a sequence of nested axis-aligned subboxes $[\vec{q}]$ of $[\vec{p}]$ which enclose the posterior feasible set $\mathcal{S}$ defined by (1). As these methods are not branch-and-bounds based, they can easily deal with high-dimensional problems. Here, ICP is used to solve the *reduction problem* for (1) which consists in finding a subbox of $[\vec{p}]$ as small as possible which encloses $\mathcal{S}$, without bisections. ICP is based on the notion of inclusion function and solution function briefly recalled in the following subsections.

2.1 Inclusion function

Let $f : R^n \to R$ be a function. An *inclusion function* is a set-valued function, denoted by $[f]$, which associates to any axis-aligned box (box for short) $[\vec{x}]$ of R^n an interval denoted $[f]([\vec{x}])$ that contains the image of f over $[\vec{x}]$. Interval analysis (Moore, 1966; Moore, 1979, Hansen, 1992; Hammer *et al.*, 1995) provides efficient tools to compute inclusion functions. If for all box $[\vec{x}]$, $[f]([\vec{x}])$ is the smallest interval that contains the image set, $[f]$ is said to be the *minimal*.

2.2 Reduction with one constraint

Let $\vec{p} = (p_1, p_2, \ldots, p_n)^{\mathrm{T}}$ be a vector of R^n. A *constraint* is a subset $\mathcal{P}_1$ of R^n (see, Benhamou and Older, 1997). Here, we shall consider only constraints of the form

$$\mathcal{P}_1 = \{\vec{p} \in R^n | f(\vec{p}) \in [y]\} = f^{-1}([y]), \tag{2}$$

where $[y]$ is an interval and f is a continuous function mapping R^n into R. Let $[\vec{p}]$ be a box, the reduction problem consists in finding a box $[\vec{z}]$, if possible small and without bisections, that contains the set

$$\mathcal{S}_1 = f^{-1}([y]) \cap [\vec{p}]. \tag{3}$$

A possible approach is based on the following theorem.

Theorem 1: Assume that it is possible to isolate p_i in the expression $f(\vec{p}) = y$, *i.e.,* there exists

a function g_i that satisfies

$$f(\vec{p}) = y \Leftrightarrow p_i = g_i({}^i\vec{p}, y), \tag{4}$$

where ${}^i\vec{p} = (p_1, \ldots, p_{i-1}, p_{i+1}, \ldots, p_n)^{\mathrm{T}}$. The function g_i is called *solution function associated with* p_i. Denote by π_i the projection operator on the ith axis and by $[g_i]$ an inclusion function for g_i. We have

$$\pi_i(\mathcal{S}_1) \subset [g_i]\left({}^i[\vec{p}]\,, [y]\right) \cap [p_i] \tag{5}$$

where ${}^i[\vec{p}] = \left([p_1]\,, \ldots, [p_{i-1}]\,, [p_{i+1}]\,, \ldots, [p_n]\right)^{\mathrm{T}}$. Moreover, if g_i is continuous and if $[g_i]$ is minimal, then the inclusion relation (5) becomes an equality. ♢

Proof: Since the following equivalences hold

$$\begin{array}{rcl} f(p_1, \ldots, p_n) \in [y] & \Leftrightarrow & (\exists y \in [y]\,|f(p_1, \ldots, p_n) = y) \\ & \overset{(4)}{\Leftrightarrow} & \left(\exists y \in [y]\,|p_i = g_i({}^i\vec{p}, y)\right), \end{array} \tag{6}$$

we have

$$\begin{array}{rcl} \pi_i(\mathcal{S}_1) & = & \{p_i \in [p_i] | \forall k \neq i, \exists p_k \in [p_k], f(p_1, \ldots, p_n) \in [y]\} \\ & \overset{(6)}{=} & \{p_i \in [p_i] | \forall k \neq i, \exists p_k \in [p_k], \exists y \in [y] \text{ with } p_i = g_i({}^i\vec{p}, y)\} \\ & = & \left\{p_i \in [p_i] | p_i \in g_i({}^i[\vec{p}]\,, [y])\right\}. \end{array} \tag{7}$$

The projection set $\pi_i(\mathcal{S}_1)$ is thus included in the interval

$$\begin{array}{rcl} [\pi_i]\,(\mathcal{S}_1) & = & \left\{p_i \in [p_i] | p_i \in [g_i]\left({}^i[\vec{p}]\,, [y]\right)\right\} \\ & = & [g_i]\left({}^i\,[\vec{p}]\,, [y]\right) \cap [p_i] \end{array}$$

with $\pi_i(\mathcal{S}_1) = [\pi_i]\,(\mathcal{S}_1)$ if $[g_i]$ is minimal and g_i is continuous. ♢

Example 1: Consider the constraint $p_1 p_2 \in [8, 40]$ and the box $[\vec{p}] = [1, 4] \times [1, 4]$. An enclosure of $\mathcal{S}_1$ defined by (3) can be obtained by reducing $[\vec{p}]$ as follows. Since

$$p_1 p_2 = y \Leftrightarrow p_1 = \frac{y}{p_2} \Leftrightarrow p_2 = \frac{y}{p_1} \text{ (for } p_i \neq 0), \tag{8}$$

the solution functions are $g_1(p_2, y) = \frac{y}{p_2}$ and $g_2(p_1, y) = \frac{y}{p_1}$. From Theorem 1, the projections of $\mathcal{S}_1$ are given by

$$\begin{array}{rcl} \pi_1(\mathcal{S}_1) & = & [g_1]\left([p_2]\,, [y]\right) \cap [p_1] = \dfrac{[y]}{[p_2]} \cap [p_1] = \dfrac{[8, 40]}{[1, 4]} \cap [1, 4] = [2, 4] \\ \pi_2(\mathcal{S}_1) & = & [g_2]\left([p_1]\,, [y]\right) \cap [p_2] = \dfrac{[y]}{[p_1]} \cap [p_2] = [2, 4], \end{array}$$

and thus, the interval hull of $\mathcal{S}_1$ is $[\mathcal{S}_1] = [2, 4] \times [2, 4]$. ♢

2.3 Reduction with m constraints

The reduction problem associated with (1) amounts to find, without branching, a subbox $[\vec{z}]$ (as small as possible) of $[\vec{p}]$ such that $\mathcal{S} \subset [\vec{z}]$. The interval $[p_i]$ is *consistent* with the constraint $\mathcal{P}_j = f_j^{-1}([y_j])$, $j \in \{1, \ldots, m\}$, if

$$[p_i] = \pi_i(\mathcal{S}_j), \tag{9}$$

where $\mathcal{S}_j = \mathcal{P}_j \cap [\vec{p}]$. The interval $[p_i]$ is *locally consistent* (also called *arc-consistent* in the literature) with (1), if it is *consistent* with all $\mathcal{S}_j$, *i.e.*,

$$[p_i] = \bigcap_{j \in \{1,\ldots,m\}} \pi_i(\mathcal{S}_j). \tag{10}$$

$[p_i]$ is *globally consistent* with (1), if

$$[p_i] = \pi_i \left(\bigcap_{j \in \{1,\ldots,m\}} \mathcal{S}_j \right) = \pi_i(\mathcal{S}). \tag{11}$$

Note that, since for two sets A and B, $\pi_i(A \cap B) \subset \pi_i(A) \cap \pi_i(B)$, the global consistency implies the local consistency. The box $[\vec{p}]$ is *globally consistent* with (1) if all its components are globally consistent with (1). In this case, $[\vec{p}]$ is the interval hull of $\mathcal{S}$.

Assume that for some $j \in \{1, \ldots, m\}$ and $i \in \{1, \ldots, n\}$, the variable p_i can be isolated in the expression $f_j(\vec{p}) = y_i$, *i.e.*, there exists a solution function g_i^j that satisfies

$$f_j(\vec{p}) = y_j \Leftrightarrow p_i = g_i^j({}^i\vec{p}, y_j). \tag{12}$$

From Theorem 1

$$\pi_i(\mathcal{S}_j) \subset [g_i^j]({}^i[\vec{p}], [y_j]) \cap [p_i]. \tag{13}$$

Therefore, $\forall j \in \{1, \ldots, m\}, \forall i \in \{1, \ldots, n\}, \pi_i(\mathcal{S}_j)$ is inside the interval

$$[h_i^j]([\vec{p}], [y_j]) = \begin{cases} [g_i^j]({}^i[\vec{p}], [y_j]) \cap [p_i] & \text{if } g_i^j \text{ exists} \\ [p_i] & \text{otherwise.} \end{cases} \tag{14}$$

The algorithm to be presented generates a nested sequence of subboxes of $[\vec{p}]$, which contains $\mathcal{S}$. This algorithm is a simple version of the *local Waltz filtering algorithm* initially presented by Waltz (1975) for real numbers and extended to intervals by Davis (1987) and Cleary (1987). The stopping criterion to be used is based on the *relative remoteness* $r([\vec{x}], [\vec{y}])$ of a box $[\vec{x}] \subset R^n$ to a box $[\vec{y}] \subset R^n$, defined by

$$r([\vec{x}], [\vec{y}]) = \max_{i \in 1,\ldots,n} \max_{x_i \in [x_i]} \max \left\{ 0, \frac{x_i - y_i^+}{|y_i^+|}, \frac{y_i^- - x_i}{|y_i^-|} \right\}. \tag{15}$$

When the reduction produced at a given iteration is smaller than a given small positive real number η, the algorithm stops.

WALTZ($[\vec{p}]$, $[\vec{y}]$)

Input: $\vec{f}$, $[\vec{p}]$, $[\vec{y}]$;
Repeat
  $[\vec{p}_0] = [\vec{p}]$;
  For $j = 1$ to m do
    For $i = 1$ to n do
      $[p_i] = [h_i^j]([\vec{p}], [y_j])$;
Until $r([\vec{p}_0], [\vec{p}]) < \eta$;
Output $[\vec{p}]$;

This algorithm is said to be *local,* because each constraint is considered independently. For a *global* approach, see (Hyvonen, 1992). We have the following properties (Benhamou and Granvilliers, 1997): WALTZ terminates if $\eta > 0$, it is *correct* (no solution is lost), if $\eta = 0$ and if the minimal inclusion functions are available for all solution functions then $[\vec{p}]$ tends toward a box which is locally consistent.

Example 2: For the set inversion problem

$$\begin{cases} p_1 + 2p_2 \in [-1, 1] \\ p_1 - p_2 \in [-1, 1] \\ \vec{p} \in [-10, 10] \times [-10, 10]. \end{cases} \tag{16}$$

the solution functions are

$$\begin{array}{ll} g_1^1(p_2, y) = y_1 - 2p_2 & g_2^1(p_1, y) = \frac{y_1 - p_1}{2} \\ g_1^2(p_2, y) = y_2 + p_2 & g_2^2(p_1, y) = -y_2 + p_1 \end{array} \tag{17}$$

The results of the first iteration of the *repeat-until* loop are given by

$$\begin{aligned} [p_1] &= ([-1, 1] - 2[-10, 10]) \cap [-10, 10] = [-10, 10] \\ [p_2] &= \left(\tfrac{[-1,1]-[-10,10]}{2}\right) \cap [-10, 10] = [-\frac{11}{2}, \frac{11}{2}] \\ [p_1] &= \left([-1, 1] + [-\frac{11}{2}, \frac{11}{2}]\right) \cap [-10, 10] = [-\frac{13}{2}, \frac{13}{2}] \\ [p_2] &= \left([-1, 1] + [-\frac{13}{2}, \frac{13}{2}]\right) \cap [-\frac{11}{2}, \frac{11}{2}] = [-\frac{11}{2}, \frac{11}{2}] \end{aligned}$$

The two constraints of (16) are represented by the hatched fields in Figure 1. The lightgrey rectangle represents the reduced box $[\vec{z}] = [-\frac{13}{2}, \frac{13}{2}] \times [-\frac{11}{2}, \frac{11}{2}]$ obtained after the first iteration. The darkgrey box is obtained after two iterations. The small white box in the center is globally consistent with (16).

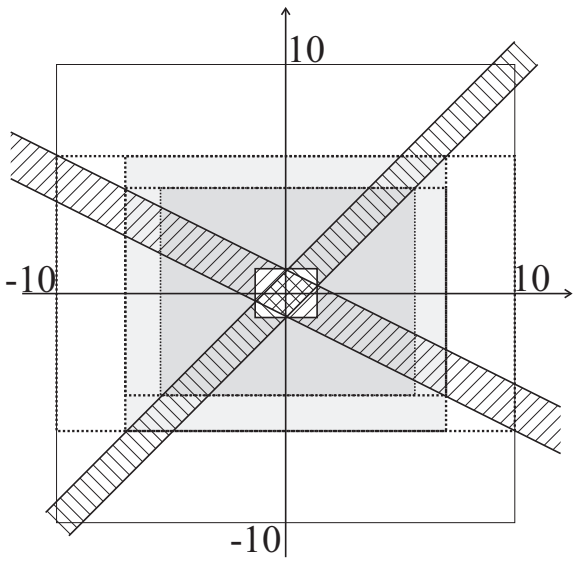

Figure 1: The two first steps of the local Waltz filtering algorithm

The procedure may stop, for instance, when each constraint contains two opposite vertices of the box $[\vec{z}]$. In the set inversion problem

$$\begin{cases} p_1 + p_2 \in [-1,1] \\ p_1 - p_2 \in [-1,1] \\ \vec{p} \in [-10,10] \times [-10,10], \end{cases} \tag{18}$$

since the hull box of $f_1^{-1}([-1,1]) \cap [\vec{p}]$ and the hull box of $f_2^{-1}([-1,1]) \cap [\vec{p}]$ are both equal to $[\vec{p}]$, the box $[\vec{p}]$ is locally consistent but not globally consistent with (18). The WALTZ algorithm is then unable to reduce $[\vec{p}]$. $\diamondsuit$

Remark 1: When carefully chosen, addition of redundant constraints drastically improve the performances of the WALTZ algorithm (see Benhamou and Granvilliers, 1997). Redundant constraints can be obtained by some elementary algebraic manipulations. In (18), by adding the two constraints, we get a third constraint given by $2p_1 \in [-2,2]$. The Waltz algorithm is now able to reduce $[\vec{p}]$. If possible, the algebraic manipulations have to be performed in order to create new constraints that involve a small number of variables. Recently, several authors have studied various combinations of solvers in the case of continuous real constraints and in particular, combinations of techniques from computer algebra and ICP (see *e.g.* Marti and Rueher, 1995). $\diamondsuit$

3 SET INVERSION WITH REDUCTION

The set inversion algorithm SIVIA($[\vec{p}]$) (Jaulin and Walter, 1993) makes it possible to enclose the set $\mathcal{S}$ given by (1) between an inner subpaving $\mathcal{S}^-$ and an outer subpaving $\mathcal{S}^+ = \mathcal{S}^- \cup \Delta\mathcal{S}$.

Recall that a subpaving is a union of nonoverlapping boxes. A new version of SIVIA that includes the local Waltz filtering procedure is now proposed. In what follows,

$$\text{width}([\vec{p}]) = \max_{i\in\{1,\ldots,n\}} \alpha_i \left(p_i^+ - p_i^-\right), \tag{19}$$

where the α_i's are arbitrary weighting coefficients. The bisections are performed with respect to the direction i that maximizes $\alpha_i \left(p_i^+ - p_i^-\right)$. The required accuracy ε is the width beyond which bisections are not allowed. For more information see (Jaulin and Walter, 1993).

SIVIA($[\vec{p}]$)
Step 1 If $[\vec{f}]([\vec{p}]) \subset [\vec{y}]$, $\{\mathcal{S}^- = \mathcal{S}^- \cup [\vec{p}]$; return$\}$;
Step 2 $[\vec{p}] =$ WALTZ$(\vec{f}, [\vec{p}], [\vec{y}])$;
Step 3 If $[\vec{p}]$ is empty, return;
Step 4 If width$([\vec{p}]) < \varepsilon$, $\{\Delta\mathcal{S} = \Delta\mathcal{S} \cup [\vec{p}]$; return$\}$;
Step 5 Bisect $[\vec{p}]$ getting the two boxes $[\vec{p}]_1$ and $[\vec{p}]_2$;
Step 6 SIVIA$([\vec{p}]_1)$; SIVIA$([\vec{p}]_2)$;

Note that if we remove the Step 2 of this algorithm, we obtain the classical SIVIA. As an application, consider a model given by a sum of exponential functions. The set inversion problem to be solved is given by (1), with

$$\begin{aligned}
[\vec{p}] &= [2,60]\times[0,1]\times[-30,-1]\times[0,0.5]\,;\\
\vec{t} &= (0.75,1.5,2.25,3,6,9,13,17,21,25)^{\mathrm{T}};\\
\vec{y} &= (7.39,4.09,1.74,0.097,-2.55,-2.69,-2.07,-1.44,-0.98,-0.66)^{\mathrm{T}}\,;\\
[y_j] &= y_j + [-0.05\ \text{abs}(y_j) - 0.1, 0.05\ \text{abs}(y_j) + 0.1]\,;\\
f_j(\vec{p}) &= p_1 e^{-p_2 t_j} + p_3 e^{-p_4 t_j};
\end{aligned}$$

Since all variables can be isolated in each constraint, the h_i^j's used by the Waltz procedure are given by

$$h_1^j = \left(y_j - p_3 e^{-p_4 t_j}\right) e^{p_2 t_j}, \tag{20}$$

$$h_2^j = \frac{-1}{t_j} \log\left(\frac{y_j - p_3 e^{p_4 t_j}}{p_1}\right), \tag{21}$$

$$h_3^j = \left(y_j - p_1 e^{-p_2 t_j}\right) e^{p_4 t_j}, \tag{22}$$

$$h_4^j = \frac{-1}{t_j} \log\left(\frac{y_j - p_1 e^{p_2 t_j}}{p_3}\right). \tag{23}$$

The weighting coefficients are chosen as

$$\alpha_1 = 0.02, \alpha_2 = 1, \alpha_3 = 0.03, \alpha_4 = 2. \tag{24}$$

For $\varepsilon = 0.2, \eta = 0.001$, the subpaving $\mathcal{S}^+ = \mathcal{S}^- \cup \Delta\mathcal{S}$ has been obtained after 15 seconds on a Pentium-133Mhz Personal Computer. The projections of $\mathcal{S}^+$ on the (p_1, p_3) and (p_2, p_4)-spaces,

are represented in Figures 2-a and 2-b. Without the reduction procedure, for $\varepsilon = 0.2$, SIVIA obtains a subpaving which goes far beyond the initial box and for $\varepsilon = 0.1$, SIVIA generates the subpaving of Figures 2-c and 2-d after 8 minutes. Figure 2, shows that even using a rough precision, SIVIA with the reduction procedure provides better results than the classical SIVIA.

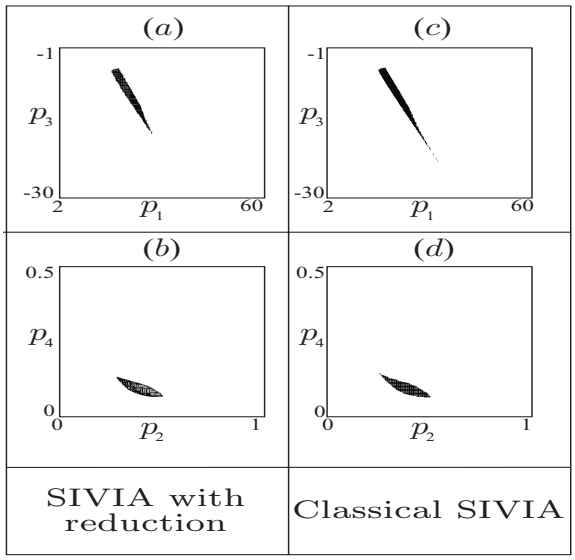

Figure 2: Comparison between the new and the former SIVIA.

Remark 2: For this testcase, each constraint involves four parameters. As illustrated by Remark 1, the performances of Waltz could be increased by adding new constraints involving less parameters. For instance, from the two equations

$$\begin{cases} p_1 e^{-p_2 t_j} + p_3 e^{-p_4 t_j} &= y_j \\ p_1 e^{-p_2 t_k} + p_3 e^{-p_4 t_k} &= y_k \end{cases} \tag{25}$$

we can get 540 solution functions of the form

$$p_i = g^{j,k}_{i,i_1,i_2}(p_{i_1}, p_{i_2}), \text{ with } i \neq i_1 \neq i_2 \text{ and } j \neq k. \tag{26}$$

One of them is

$$p_2 = g^{j,k}_{1,3,4}(p_3, p_4) = \frac{1}{t_j - t_k}\left(\log\left(y_k - p_3 e^{-p_4 t_k}\right) - \log\left(y_j - p_3 e^{-p_4 t_j}\right)\right). \tag{27}$$

These solution functions can be inserted in Waltz, but only few of them would improve the efficiency of the method. If all the $g^{j,k}_{i,i_1,i_2}$ are considered in Waltz, the computing time of an iteration becomes too high. The choice of those which are worth to be inserted is often very difficult (see Benhamou and Grandvilliers (1997) in the case where the f_j's are polynomial). For the sake of simplicity, this possible improvement of the method has not been considered in the application considered in this paper. $\diamondsuit$

4 COMPUTING THE INTERVAL HULL

In many situations, we are not interested in an accurate representation of the posterior feasible set $\mathcal{S}$, given by (1), but only in the smallest box (or interval hull) $[\mathcal{S}]$ of $\mathcal{S}$. The components of $[\mathcal{S}]$ represent the feasible intervals for the parameters and the center of $[\mathcal{S}]$ is a point estimate which enjoys useful optimality properties (Milanese *et al.*, 1986). The SIVIA algorithm presented in the previous section can easily be transformed to compute a bracketing of $[\mathcal{S}]$, but, as the generated subpavings accumulate on the whole frontier of $\mathcal{S}$, many unnecessary computations are performed. In this section, we propose a new algorithm which brackets $[\mathcal{S}]$ between two boxes $[\vec{s}_{\rm in}]$ and $[\vec{s}_{\rm out}]$, *i.e.,*

$$[\vec{s}_{\rm in}] \subset [\mathcal{S}] \subset [\vec{s}_{\rm out}]\,. \tag{28}$$

It is assumed that the solution functions associated with each p_i are available, *i.e.,* for all $i \in \{1,\ldots,n\}$ and $j \in \{1,\ldots,m\}$, there exists a function $g_i^j({}^i\vec{p}, y_j)$ such that

$$f_j(\vec{p}) = y_j \Leftrightarrow p_i = g_i^j({}^i\vec{p}, y_j). \tag{29}$$

4.1 Cross propagation algorithm

The aim of the cross propagation algorithm to be presented is to find a box $[\vec{s}_{\rm in}]$, as large as possible, which is included in $[\mathcal{S}]$. The principle of the approach is not new: it consists in finding some feasible $\vec{p}$; the interval hull of all these feasible $\vec{p}$ is thus an approximation of $[\mathcal{S}]$. What is new is that interval analysis is used to increase the efficiency of the local researches. The local research of some feasible points in the ith direction is based on the following theorem.

Theorem 2: Let $\vec{q}$ be a point in R^n. Denote by $\mathcal{D}_i(\vec{q})$, the line, parallel to the ith axis, which contains $\vec{q}$. The intersection between $\mathcal{D}_i(\vec{q})$ and $\mathcal{S}$ is given by

$$\lambda_i(\vec{q},\mathcal{S}) = \left(q_1,\ldots,q_{i-1},\bigcap_{j=1}^{m} g_i^j({}^i\vec{q},[y_j]) \cap [p_i], q_{i+1},\ldots,q_n \right)$$

. ♢

Proof: $\vec{z} \in \lambda_i(\vec{q},\mathcal{S})$ is equivalent to

$$\left\{ \begin{array}{c} \vec{z} \in \mathcal{D}_i(\vec{q}) \\ z_i \in \bigcap_{j=1}^{m} g_i^j({}^i\vec{q},[y_j]) \cap [p_i], \end{array} \right.$$

i.e.,

$$\left\{ \begin{array}{c} \vec{z} \in \mathcal{D}_i(\vec{q}) \\ \forall j \in \{1,\ldots,m\}, \exists y_j \in [y_j] | z_i = g_i^j({}^i\vec{q}, y_j) \text{ and } z_i \in [p_i], \end{array} \right. \tag{30}$$

which is equivalent to $\vec{z} \in \mathcal{D}_i(\vec{q}) \cap \mathcal{S}$. $\diamondsuit$

Remark 3: A consequence of this theorem is that if $i \in \{1, \ldots, n\}$,

$$(\mathcal{D}_i(\vec{q}) \cap \mathcal{S}) \subset [\lambda_i](\vec{q}, \mathcal{S}) = \left(q_1, \ldots, q_{i-1}, \bigcap_{j=1}^{m} [g_i^j]({}^i\vec{q}, [y_j]) \cap [p_i], q_{i+1}, \ldots, q_n \right)^{\mathrm{T}}. \quad (31)$$

Note that $[\lambda_i](\vec{q}, \mathcal{S})$ is a degenerated box. This inclusion becomes an equality if g_i^j is continuous and $[g_i^j]$ is the minimal. $\diamondsuit$

The following recursive algorithm CROSS($\vec{q}$) assumes that minimal inclusion functions are available for all g_i^j's. $[\vec{s}_{\text{in}}]$ is a global variable which represents a (possibly empty) subbox of $[\mathcal{S}]$. When a feasible point $\vec{q}$ is found, $[\vec{s}_{\text{in}}]$ is increased in order to contain $\vec{q}$. In CROSS($\vec{q}$), $[\vec{s}_{\text{in}}] \sqcup \vec{z}$ denotes the smallest box which contains $[\vec{s}_{\text{in}}]$ and $\vec{z}$, $\vec{z}^{-}$ is the corner of $[\vec{z}]$ whose components are the lower bounds of the interval components of $[\vec{z}]$ and $\vec{z}^{+}$ is the corner of $[\vec{z}]$ at the opposite of $\vec{z}^{-}$.

CROSS($\vec{q}$)

Step 1 For $i = 1$ to n
Step 2 $\quad [\vec{z}] = [\lambda_i](\vec{q}, \mathcal{S})$;
Step 3 $\quad$ if $[\vec{z}] = \emptyset$, next i;
Step 4 $\quad$ if $r(\vec{z}^{-}, [\vec{s}_{\text{in}}]) > \kappa$, $[\vec{s}_{\text{in}}] = [\vec{s}_{\text{in}}] \sqcup \vec{z}^{-}$; CROSS($\vec{z}^{-}$);
Step 5 $\quad$ if $r(\vec{z}^{+}, [\vec{s}_{\text{in}}]) > \kappa$, $[\vec{s}_{\text{in}}] = [\vec{s}_{\text{in}}] \sqcup \vec{z}^{+}$; CROSS($\vec{z}^{+}$);
Step 6 Next i;

When CROSS($\vec{q}$) is called by the main program (HULL, given in the next subsection), $[\vec{s}_{\text{in}}]$ is a (possibly empty) box which has been proved to be included in $[\mathcal{S}]$. The *for* loop generates a set of segments (the cross) that belong to $\mathcal{S}$. Each vertex $\vec{z}^{-}$ and $\vec{z}^{+}$ of each segment of the cross is feasible and is thus likely to increase $[\vec{s}_{\text{in}}]$. If $[\vec{s}_{\text{in}}]$ is increased significantly, at Steps 4 and 5, an other call of CROSS occurs.

4.2 Bracketing the interval hull

The following recursive algorithm HULL($[\vec{p}]$) generates two boxes $[\vec{s}_{\text{in}}]$ and $[\vec{s}_{\text{out}}]$ that satisfy (28). From a computer point of view, $[\vec{s}_{\text{in}}]$ and $[\vec{s}_{\text{out}}]$ are two global variables initialized as: $[\vec{s}_{\text{in}}] = \emptyset$ and $[\vec{s}_{\text{out}}] = \emptyset$. After completion of the algorithm, the precision of the bracketing provided by HULL is expected to satisfy,

$$r([\vec{s}_{\text{out}}], [\vec{s}_{\text{in}}]) \leq \nu. \quad (32)$$

Most often, when ε is small enough, (32) is satisfied, but there exists some atypical situations in which it not true. In what follows, $[\vec{s}_{\text{out}}] \sqcup [\vec{s}_{\text{in}}]$ denotes the interval hull of $[\vec{s}_{\text{out}}] \cup [\vec{s}_{\text{in}}]$.

$\boxed{\text{HULL}([\vec{p}])}$

Step 1 CROSS (Center ($[\vec{p}]$)) ; $[\vec{s}_{\text{out}}] = [\vec{s}_{\text{out}}] \sqcup [\vec{s}_{\text{in}}]$;
Step 2 $[\vec{p}] = \text{WALTZ}(\vec{f}, [\vec{p}], [\vec{y}])$;
Step 3 If $[\vec{p}]$ is empty, return;
Step 4 If width($[\vec{p}]$) $< \varepsilon$ or $r([\vec{p}], [\vec{s}_{\text{in}}]) < \nu$, $\{[\vec{s}_{\text{out}}] = [\vec{s}_{\text{out}}] \sqcup [\vec{p}]$; return$\}$;
Step 5 Bisect $[\vec{p}]$ getting the two boxes $[\vec{p}]_1$ and $[\vec{p}]_2$;
Step 6 HULL($[\vec{p}]_1$); HULL($[\vec{p}]_2$);

At Step 1, CROSS attempts to increase $[\vec{s}_{\text{in}}]$ by using a local approach to find some new feasible $\vec{p}$. The current box is reduced at Step 3. The ε-condition at Step 4 assures that HULL is a finite algorithm. The ν-condition is due to (32) and avoids unnecessary refinement.

4.3 Application

Consider the testcase presented in Section 3. For $\varepsilon = \eta = \kappa = 0.001$ and $\nu = 0.01$, HULL finds in less than 8 seconds on a Pentium-133 MHz Personal Computer (speed: 13 Mflops/sec) that

$$[\vec{s}_{\text{in}}] = [17.2, 26.79] \times [0.301, 0.49] \times [-16, -5.4] \times [0.0767, 0.1354] \tag{33}$$

$$[\vec{s}_{\text{out}}] = [17.05, 27] \times [0.298, 0.495] \times [-16.2, -5.34] \times [0.0763, 0.1359] \tag{34}$$

In Milanese and Vicino (1991) a signomial approach is proposed to solve this problem. The algorithm of Falk (1973) is used and it obtains in 10 minutes on a VAX 8800 (speed: 1.3 Mflops/sec) that $[\mathcal{S}]$ is approximated, with a relative error of 2 percents, by the box,

$$[\vec{p}_{\text{MV}}] = [17.2, 26.9] \times [0.3, 0.49] \times [-16.1, -5.4] \times [0.077, 0.136] \tag{35}$$

which corresponds to the bracketing

$$[\vec{s}_{\text{in}}^{\text{MV}}] = [17.55, 26.36] \times [0.306, 0.48] \times [-15.77, -5.51] \times [0.0786, 0.133], \tag{36}$$

$$[\vec{s}_{\text{out}}^{\text{MV}}] = [16.85, 27.44] \times [0.294, 0.5] \times [-16.43, -5.29] \times [0.0754, 0.1388]. \tag{37}$$

Note that

$$[\vec{s}_{\text{in}}^{\text{MV}}] \subset [\vec{s}_{\text{in}}] \subset [\mathcal{S}] \subset [\vec{s}_{\text{out}}] \subset [\vec{s}_{\text{out}}^{\text{MV}}] \tag{38}$$

The results obtained by HULL are thus more accurate and more efficient (about 8 times faster if the computations are performed on the same machine) than those obtained by Milanese and Vicino (1991).

Remark 4: As presented above, the class of problems that can be treated by HULL is rather limited because all solution functions and their associated minimal inclusion functions are supposed to be available. When these assumptions are not fulfilled, HULL could be adapted for instance by replacing the interval local research by a classical one. Concerning the Falk algorithm, it assumes that the f_i's are polynomial.

5 CONCLUSION

In this paper, interval constraint propagation (ICP) and some of its applications to bounded-error estimation is presented. In the first part of the paper, ICP is used to increase the efficiency of the set-inversion algorithm SIVIA. Using an example from the literature, the improved version of SIVIA is compared to the former SIVIA. It obtains more accurate results in a 30 times shorter computing time. A new algorithm for obtaining the interval hull of the posterior feasible set is presented in the second part of the paper. This algorithm alternates an ICP-based elimination procedure with a new interval-based procedure that performs local researches of some feasible points. By an example taken from the literature, this algorithm has been shown to be more efficient than the signomial-based algorithm of Falk (1973).

When dealing with interval-based branch-and-bound algorithm, the use of reducing methods makes it possible to decrease drastically the number of bisections, thus allowing much more efficient algorithms. ICP provides a simple way to reduce the boxes, but, as it are local, it is unable to take into account the dependences between constraints. Therefore some bisections that could be avoided have to be done if only local propagation techniques are considered. Other reduction techniques which are more global, *i.e.,* which take into account the dependences between constraints, will be studied in ongoing researches. We quote two of them: the automatic generation of redundant constraints (see Benhamou and Granvilliers, 1997) and the interval Newton reduction method (Moore, 1966; Hansen, 1995).

Notation Table

π_i	:	projection operator on the ith axis.
$\lambda_i(\vec{q}, \mathcal{S})$	:	intersection between the line $\mathcal{D}_i(\vec{q})$ and $\mathcal{S}$.
ε	:	minimum width of boxes allowed to be bisected in SIVIA and HULL.
η	:	minimum progress required in WALTZ.
κ	:	minimum progress required in CROSS.
ν	:	required remoteness between the inner and outer boxes in HULL.
${}^i\vec{p}$	:	$(p_1, \ldots, p_{i-1}, p_{i+1}, \ldots, p_n)^{\mathrm{T}}$.
$\mathcal{D}_i(\vec{q})$	:	line of R^n containing $\vec{q}$ and parallel to the ith axis.
$\mathcal{S}$	:	posterior feasible set, see relation (1).
$r([\vec{x}], [\vec{y}])$	:	remoteness of $[\vec{x}]$ to $[\vec{y}]$, see equation (15).
$[\mathcal{S}]$	:	interval hull of $\mathcal{S}$, *i.e.*the smallest axis-aligned box which encloses $\mathcal{S}$.
$\mathcal{A} \sqcup \mathcal{B}$	:	interval hull of $\mathcal{A} \cup \mathcal{B}$.

REFERENCES

Benhamou, F. and L. Granvilliers (1997). Automatic generation of numerical redundancies for non-linear constraint solving. *Reliable Computing*, **3**, 335-344.

Benhamou, F. and W. Older (1997). Applying interval arithmetic to real, integer and boolean constraints. *Journal of Logic Programming*, **32**, 1-24.

Belforte G., B. Bona and V. Cerone (1990). Parameter estimation algorithms for a set membership description of uncertainty. *Automatica,* **26**, 887-898.

Cleary, J. C. (1987). Logical arithmetic. *Future Computing Systems,* **2**, (2), 125-149.

Davis, E. (1987). Constraint propagation with interval labels. *Artificial Intelligence*, **32**, 281-331.

Falk, J. E. (1973). Global solutions for signomial programs. *Tech. Rep. T-274, George Washington Univ.,* Washington, DC.

Fogel, E. and Y. F. Huang (1982). On the value of information in system identification - bounded noise case. *Automatica*, **18**, 229-238.

Hammer, R., M. Hocks, U. Kulisch, and D. Ratz (1995). *C++ Toolbox For Verified Computing.* Springer-Verlag, Heidelberg.

Hansen, E. (1992). *Global Optimization Using Interval Analysis.* Marcel Dekker, New York.

Hyvönen, E. (1992). Constraint reasoning based on interval arithmetic; The tolerance propagation approach. *Artificial Intelligence,* **58**, 71-112.

Jaulin, L. and E. Walter (1993). Set inversion via interval analysis. *Automatica*, **29**, 1053-1064.

Mackworth, A. (1977). Consistency in networks of relations. *Artificial intelligence*, **8**, 99-118.

Marti, P. and M. Rueher (1995). A distributed cooperating constraint solving system. *International Journal of Artificial Intelligence Tools*, **4**, 93-113.

Milanese, M. and G. Belforte (1982). Estimation theory and uncertainty interval evaluation in presence of unknown-but-bounded errors. Linear families of models and estimators. *IEEE Trans. Aut. Control*, **27**, 408-414.

Milanese, M., R. Tempo and A. Vicino (1986). Strongly optimal algorithms and optimal infor-

mation in estimation problems. *J. Complexity,* **2**, 78-98.

Milanese, M. and A. Vicino (1991). Estimation theory for nonlinear models and set membership uncertainty. *Automatica,* **27**, 403-408.

Milanese, M., J. Norton, H. Piet-Lahanier and E. Walter (Eds.) (1996). *Bounding Approaches to System Identification.* Plenum, New York.

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

Moore, R. E. (1979). *Methods and Applications of Interval Analysis.* SIAM, Philadelphia.

Norton, J. P. (1987). Identification of parameter bounds for ARMAX models from records with bounded noise. *Int. J. Control*, **45**, 375–390.

Norton, J. P. (Ed.) (1994). Special issue on bounded-error estimation, **1**. *Int. J. Adapt. Control Signal Proc.*, **8**(1), 1–118.

Norton, J. P. (Ed.) (1995). Special issue on bounded-error estimation, **2**. *Int. J. Adapt. Control Signal Proc.*, **9**(1), 1–132.

Walter, E. and H. Piet-Lahanier (1989). Exact recursive polyhedral description of the feasible parameter set for bounded-error models. *IEEE Trans. Aut. Control,* **34**, 911-915.

Walter, E. (Ed.) (1990). Parameter identifications with error bounds. special issue of *Mathematics and Computers in Simulation*, **32**(5&6), 447-638.

Walter, E. and L. Pronzato (1997). *Identification of Parametric Models from Experimental Data.* Springer, London.

Waltz, D. L. (1975). Generating semantic descriptions from drawings of scenes with shadows. in: P.H. Winston, editor, *The Psychology of Computer Vision*, McGraw-Hill, New York, 19-91.