Guaranteed robust nonlinear minimax estimation

Luc Jaulin, Eric Walter

Guaranteed Robust Nonlinear Minimax Estimation

L. Jaulin [1,2] and E. Walter[1]

Abstract: Minimax parameter estimation aims at characterizing the set of all values of the parameter vector that minimize the largest absolute deviation between experimental data and corresponding model outputs. However, minimax estimation is well known to be extremely sensitive to outliers in the data resulting, *e.g.*, of sensor failures. In this paper, a new method is proposed to robustify minimax estimation by allowing a prespecified number of absolute deviations to become arbitrarily large without modifying the estimates. By combining tools of interval analysis and constraint propagation, it becomes possible to compute the corresponding minimax estimates in an approximate but guaranteed way, even when the model output is nonlinear in its parameters. The method is illustrated on a problem where the parameters are not globally identifiable, which demonstrates its ability to deal with the case where the minimax solution is not unique.

Keywords: constraint propagation, interval computation, nonlinear estimation, minimax estimation, outliers, robust estimation.

1 Introduction

When the parameter vector $\mathbf{p}$ of a model has to be estimated from experimental data $\mathbf{y}$, the procedure to be followed depends on the assumptions about the noise. If the components of $\mathbf{y}$ are assumed independently corrupted by an additive noise uniformly distributed over the interval $[-\delta, \delta]$, with δ unknown, then a maximum-likelihood estimate of $\mathbf{p}$ is obtained by minimizing the largest absolute deviation between the data and the corresponding model outputs, which correspond to minimax estimation. The resulting estimate $\widehat{\mathbf{p}}$ belongs to the set of all parameter vectors that are consistent with any value of δ large enough for the set to be nonempty. Moreover, the corresponding largest absolute deviation $\hat{\delta}$ is a lower bound for δ, which provides useful information to anyone interested in bounded-error parameter estimation. (See [[18], [19], [21], [16], [5], [14] and the references therein).

Minimax estimation is well known, however, to be extremely sensitive to outliers, as a single of them may suffice to ruin the estimate [20], [2]. Outliers are data that result of events not accounted for by the model, such as sensor failures, transcription errors or erroneous hypotheses on noise distribution. The purpose of this paper is to present a new algorithm for computing

[1]Laboratoire des Signaux et Systèmes, UMR 8506, CNRS-Supélec-Université Paris Sud, 91192 Gif-sur-Yvette, France

[2]on leave from Laboratoire d'Ingénierie des Systèmes Automatisés, Université d'Angers, France.

guaranteed robust minimax estimates, robust meaning here that a prespecified number of absolute deviations are allowed to become arbitrarily large, and guaranteed meaning that an outer approximation of the set of robust minimax estimates is obtained. The basic idea is akin to that in [15] and [2], but its implementation is radically new. It combines the tools of *interval analysis* and *constraint propagation* to provide guaranteed results (contrary to [15]) for nonlinear models (contrary to [2]). This combination is called *Interval Constraint Propagation* (ICP) [6], [7]. Some basic notions of ICP are recalled in Section 3, and the necessity of extending it to deal with robust minimax estimation (RME) is stressed. Section 4 gives a rather classical optimization algorithm based on ICP. In order to develop an efficient reduction procedure, able to handle RME, Section 5 introduces the notion of set polynomials. To the best of our knowledge, this notion is new, at least in this context. A test-case is presented in Section 5 to demonstrate the efficiency of the approach advocated.

2 Relaxed minimax estimator

In what follows, the parameter vector $\mathbf{p} \in \mathbb{R}^n$ is assumed to belong to the prior axis-aligned search box $[\mathbf{p}_0]$. Let $\mathbf{y} \in \mathbb{R}^m$ be the data vector, $\mathbf{y}_{\mathrm{m}}(\mathbf{p}) \in \mathbb{R}^m$ be the associated model output vector and $\mathbf{f}(\mathbf{p}) \in \mathbb{R}^m$ be the (absolute) error vector defined as $|\mathbf{y} - \mathbf{y}_{\mathrm{m}}(\mathbf{p})|$, where the absolute value is taken componentwise. Denote by q the number of data allowed to become arbitrarily large. Define the q-max function from $\mathbb{R}^m \to \mathbb{R}$, where m is an integer with $m > q \geq 0$, as the function that associates to $\mathbf{x} = (x_1, \ldots, x_m)^{\mathrm{T}}$ its $(q+1)$th largest entry. For example, if $\mathbf{x} = (3, -4, 3, 5, 0)^{\mathrm{T}}$, then 0-max$(\mathbf{x}) = 5$, 1-max$(\mathbf{x}) = 3$, 2-max$(\mathbf{x}) = 3$, 3-max$(\mathbf{x}) = 0$ and 4-max$(\mathbf{x}) = -4$. In a minimax context, the cost function to be used if q outliers are assumed can be written

$$j_q(\mathbf{p}) = q\text{-max}\,(\mathbf{f}(\mathbf{p}))\,. \tag{1}$$

With any given q and $\mathbf{y}$, the *relaxed minimax estimator* (RME) associates the set

$$\widehat{\mathcal{S}}_q = \arg\min_{\mathbf{p}\in[\mathbf{p}_0]} j_q(\mathbf{p})\,, \tag{2}$$

which is often a singleton or a finite number of vectors. Note that since $j_q(\mathbf{p})$ is a decreasing function of q, the minimum $\widehat{j}_q$ of $j(\mathbf{p})$ over $[\mathbf{p}_0]$ is also a decreasing function of q, *i.e.*,

$$q_1 \leq q_2 \Leftrightarrow \hat{j}_{q_2} \leq \hat{j}_{q_1}\,. \tag{3}$$

The set

$$\mathcal{S}_q(\delta) \triangleq \{\mathbf{p} \in [\mathbf{p}_0] \mid j_q(\mathbf{p}) \leq \delta\}\,. \tag{4}$$

is increasing with q, *i.e.*, $\mathcal{S}_0(\delta) \subset \mathcal{S}_1(\delta) \subset \ldots \subset \mathcal{S}_m(\delta) = [\mathbf{p}_0]$ and with δ, *i.e.*, $\delta_1 \leq \delta_2 \Leftrightarrow \mathcal{S}_q(\delta_1) \subset \mathcal{S}_q(\delta_2)$. Figure 1 illustrates these properties when the dimension of p is 1. For readability, the dependency of the set $\mathcal{S}_q$ in δ is not mentioned.

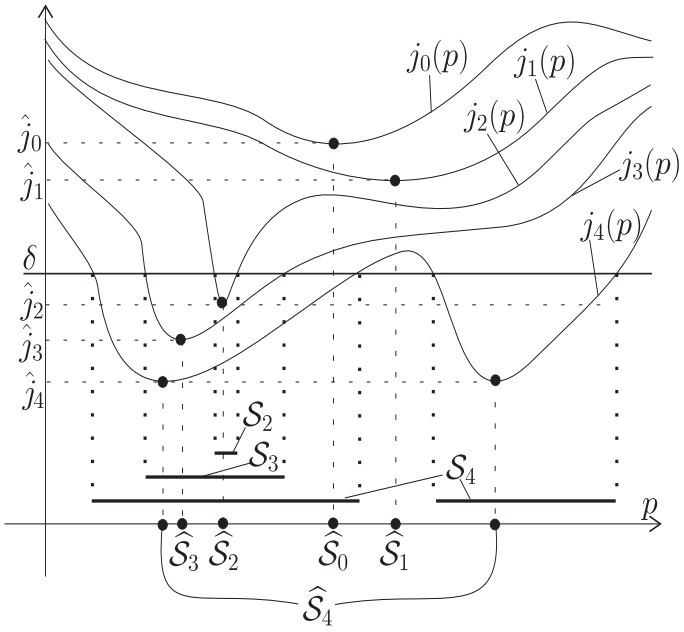

Figure 1: Illustration of the properties of the cost function: whereas $j_q(p)$ and $\hat{j}_q$ are decreasing functions of q, $\mathcal{S}_q(\delta)$ increases with q and δ

To implement *RME*, an efficient and reliable algorithm is needed to compute the set $\hat{\mathcal{S}}_q$ of all minimizers of $j_q(\mathbf{p})$ on $[\mathbf{p}_0]$. This task will be performed by the algorithm MINIMIZE, to be presented in Section 4, which is based on interval constraint propagation (ICP), briefly recalled in the next section.

3 Interval arithmetic and constraint propagation

The approach to be employed combines two complementary tools, namely interval analysis [17] and constraint propagation [22] into what is known as *interval constraint propagation* (ICP) [6], [7]. Note that interval analysis is also used for reliable global optimization without constraint propagation, see, *e.g.*, [8], [23] in a general context and [24], [13] in a minimax context. Reliable global optimization based on ICP is often more efficient [25], [9]. Moreover, ICP handles subsets of $\mathbb{R}$ (or domains) that may not be intervals. Although such domains are less easily manipulated than intervals, they allow more accurate outer approximations of sets.

3.1 Interval and domain arithmetics

A *domain* $\mathcal{X}$ of $\mathbb{R}$ is a subset of $\mathbb{R}$. *Domain arithmetic* is a generalization for domains of the classical arithmetic for real numbers. Let $\mathcal{X}$ and $\mathcal{Y}$ be two domains, $\oplus$ an operator in

$\{+, -, *, /, \hat{}, \max, \min \ldots\}$ and f a real function such as sin, cos, tan, sqr, abs...

$$\begin{array}{rcl} \mathcal{X} \oplus \mathcal{Y} & = & \{x \oplus y | x \in \mathcal{X}, y \in \mathcal{Y}\}, \\ f(\mathcal{X}) & = & \{f(x) | x \in \mathcal{X}\}. \end{array} \tag{5}$$

When the domains, to be handled are intervals, classical operations as defined by interval analysis can be used to perform these operations. Here, the domains are assumed to consist of finite unions of intervals and interval computation can be extended to computing with such domains [10]. The main advantages of using domains instead of intervals are that the set of domains is closed with respect to the union operator $\cup$ and that domain computation makes it possible to avoid *hull pessimism* when discontinuous or multivalued functions are involved ([10], [13]). For instance, with interval arithmetic $1/[-1,1] =]-\infty, \infty[$, whereas with domain arithmetic $1/[-1,1] =] - \infty, -1] \cup [1, \infty[$.

We shall call *Cartesian domain* of $\mathbb{R}^n$ the Cartesian product of n domains of $\mathbb{R}$, *i.e.*, $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$. The notion of Cartesian domain can be interpreted as an extension of that of axis-aligned box (or interval vector). This extension allows a more accurate outer bounding of compact sets with disconnected parts. Note that an axis-aligned box is a Cartesian domain, the components $\mathcal{X}_i$ of which are intervals. We shall denote the set of all Cartesian domains of $\mathbb{R}^n$ by $\mathcal{D}(\mathbb{R}^n)$. Vector calculus can be extended to Cartesian domains using interval arithmetic [17], and the notion of inclusion function. Consider a function $f : \mathbb{R}^n \to \mathbb{R}$. An *inclusion function* of f is a function $\mathcal{F} : \mathcal{D}(\mathbb{R}^n) \to \mathcal{D}(\mathbb{R})$

$$\forall \mathcal{X} \in \mathcal{D}(\mathbb{R}^n), \ f(\mathcal{X}) \subset \mathcal{F}(\mathcal{X}), \tag{6}$$

where $f(\mathcal{X}) = \{f(\mathbf{x}) | \mathbf{x} \in \mathcal{X}\}$. Domain arithmetic makes it possible to compute inclusion functions for a very large class of functions f. The principle is to replace each occurrence of a variable x_i in the expression of f by the corresponding domain and each operator or basic function by its domain counterpart, as defined by (5). The following example illustrates the computation on domains and demonstrates the pessimism resulting from multiple occurrences of variables in the expression of f.

Example 1 *Consider the function* $f(\mathbf{x}) = x_1 + x_1 x_2$, *and the domain* $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2$, *with* $\mathcal{X}_1 = [1,2]$ *and* $\mathcal{X}_2 = [-3,-2] \cup [3,4]$. *A possible inclusion function for* f *is* $\mathcal{F}(\mathcal{X}) = \mathcal{X}_1 + \mathcal{X}_1\mathcal{X}_2$ *that is evaluated as follows*

$$\begin{array}{rcll} \mathcal{F}(\mathcal{X}) & = & [1,2] + ([1,2] * ([-3,-2] \cup [3,4])) & (7) \\ & = & [1,2] + ([-6,-2] \cup [3,8]) & (8) \\ & = & [-5,0] \cup [4,10]. & (9) \end{array}$$

If $f(\mathbf{x})$ *is rewritten as* $g(\mathbf{x}) = x_1(1 + x_2)$, *a new inclusion function is obtained as* $\mathcal{G}(\mathcal{X}) =$

$\mathcal{X}_1\,(1+\mathcal{X}_2)$. *Its evaluation yields*

$$\begin{aligned}\mathcal{G}(\mathcal{X}) &= [1,2]*([1,1]+([-3,-2]\cup[3,4])) &(10)\\ &= [1,2]*([-2,-1]\cup[4,5]) &(11)\\ &= [-4,-1]\cup[4,10]. &(12)\end{aligned}$$

Note that $f(\mathcal{X})=\mathcal{G}(\mathcal{X})\subset\mathcal{F}(\mathcal{X})$, i.e., $\mathcal{G}(\mathcal{X})$ *provides the exact image of* $\mathcal{X}$ *by* f *whereas* $\mathcal{F}(\mathcal{X})$ *provides only an outer approximation (because the two occurrences of* x_1 *are treated as if they were independent).* ■

With domain computation, inclusion functions can be obtained for a large class of functions f for which an analytical expression is available. We shall see in Section 5.4 how to obtain an inclusion function for the cost function $j_q(\mathbf{p})$ defined by (1) with the help of the notion of set polynomials introduced in Section 5.

3.2 Interval constraint propagation

We shall call *primitive constraints* relations involving up to three real variables that can be written in one of the three following forms:

(unary constraint)	$z_1\in\mathcal{Z}$,	where $\mathcal{Z}\in\mathcal{D}(\mathbb{R})$,
(binary constraint)	$z_1=f(z_2)$,	where $f\in\{\cos,\sin,\exp,\log,\mathrm{sqr},\mathrm{sqrt}\dots\}$,
(ternary constraint)	$z_1=z_2\oplus z_3$,	where $\oplus\in\{+,\ -,\ *,\ /,\ \hat{}\,,\max,\min\}$.

A constraint is *and-decomposable* if it can be decomposed into a finite set of primitive constraints related by the Boolean operator *and.* For instance, the constraint

$$(\max(x_1x_2,x_1-\log(x_2)))^2+\exp(x_1)\leq 3 \tag{13}$$

is *and*-decomposable since it admits the following decomposition into primitive constraints:

$$\begin{cases} z_1=x_1x_2\\ z_2=\log(x_2)\\ z_3=x_1-z_2\\ z_4=\max(z_1,z_3)\\ z_5=\mathrm{sqr}\,(z_4)\\ z_6=\exp(x_1)\\ z_7=z_5+z_6\\ z_7\in\,]-\infty,3]. \end{cases} \tag{14}$$

If f is a function from $\mathbb{R}^n$ to $\mathbb{R}$, $\mathcal{X}\in\mathcal{D}(\mathbb{R}^n)$ and $\mathcal{Y}\in\mathcal{D}(\mathbb{R})$, ICP makes it possible to obtain, in a very efficient way, a Cartesian domain that encloses the set

$$\mathcal{S}=\mathcal{X}\cap f^{-1}(\mathcal{Y}), \tag{15}$$

provided that the constraint $\mathbf{x} \in f^{-1}(\mathcal{Y})$ is *and*-decomposable (see, *e.g.*, [10], [4], [3]). To *contract* $\mathcal{X}$ with respect to $\mathcal{S}$ means to find a Cartesian domain $\mathcal{R}$ such that $\mathcal{S} \subset \mathcal{R} \subset \mathcal{X}$. Let us illustrate, on a simple example, how ICP is used to contract a box or a Cartesian domain $\mathcal{X}$.

Example 2 *Consider the set $\mathcal{S}$ defined by (15), where $f(\mathbf{x}) \triangleq x_1^2 x_2 + x_3 x_1$, $\mathcal{X} = [1,10]^3$ and $\mathcal{Y} = [-4,4]$. To contract $\mathcal{X}$, first decompose the constraint $f(\mathbf{x}) \in \mathcal{Y}$ as*

$$\begin{array}{ll} (C1) & z_1 = x_1^2, \\ (C2) & z_2 = x_2 z_1, \\ (C3) & z_3 = x_3 x_1, \\ (C4) & y = z_2 + z_3, \\ (C5) & y \in [-4,4]. \end{array}$$

The domains to which the variables $x_1, x_2, x_3, z_1, z_2, z_3$ and y are a priori *assumed to belong are given by $[x_1] = [x_2] = [x_3] = [1,10]$, $[z_1] = [z_2] = [z_3] = [y] =]-\infty, \infty[$. For simplicity, these domains have been chosen as intervals, but this is not required by the method. By propagating these four constraints as long as contraction takes place, one gets*

$$\begin{array}{lllll} (C5) & \rightarrow & [y] := [-4,4] & & \\ (C1) & \rightarrow & [z_1] := [z_1] \cap ([x_1])^2 & = ([1,10])^2 & = [1,100]\,, \\ (C2) & \rightarrow & [z_2] := [z_2] \cap ([x_2] * [z_1]) & = [1,10] * [1,100] & = [1,1000]\,, \\ (C3) & \rightarrow & [z_3] := [z_3] \cap ([x_3] * [x_1]) & = [1,10] * [1,10] & = [1,100]\,, \\ (C4) & \rightarrow & [y] := [y] \cap ([z_2] + [z_3]) & = [-4,4] \cap ([1,1000] + [1,100]) & = [2,4], \\ (C4) & \rightarrow & [z_2] := [z_2] \cap ([y] - [z_3]) & = [1,1000] \cap ([2,4] - [1,100]) & = [1,3]\,, \\ (C4) & \rightarrow & [z_3] := [z_3] \cap ([y] - [z_2]) & = [1,100] \cap ([1,4] - [2,3]) & = [1,3]\,, \\ (C3) & \rightarrow & [x_1] := [x_1] \cap ([z_3] \,/\, [x_3]) & = [1,10] \cap ([1,3] \,/\, [1,10]) & = [1,3]\,, \\ (C3) & \rightarrow & [x_3] := [x_3] \cap ([z_3] \,/\, [x_1]) & = [1,10] \cap ([1,3] \,/\, [1,3]) & = [\mathbf{1},\mathbf{3}]\,, \\ (C2) & \rightarrow & [z_1] := [z_1] \cap ([z_2] \,/\, [x_2]) & = [1,100] \cap ([1,3] \,/\, [1,10]) & = [1,3]\,, \\ (C2) & \rightarrow & [x_2] := [x_2] \cap ([z_2] \,/\, [z_1]) & = [1,10] \cap ([1,3] \,/\, [1,3]) & = [\mathbf{1},\mathbf{3}]\,, \\ (C1) & \rightarrow & [x_1] := [x_1] \cap \sqrt{[z_1]} & = [1,3] \cap \sqrt{[1,3]} & = \left[\mathbf{1},\sqrt{\mathbf{3}}\right]. \end{array} \tag{16}$$

Thus,

$$\left\{ \mathbf{x} \in [1,10]^3 \mid x_1^2 x_2 + x_3 x_1 \in [-4,4] \right\} \subset \left[1, \sqrt{3}\right] \times [1,3]^2 \,. \tag{17}$$

■

When the constraints encountered are not *and*-decomposable, the classical ICP approach does not apply directly. For instance, the constraint $j_q(\mathbf{p}) \leq j^+$ involved in the solution of the robust minimax estimation problem is not *and*-decomposable, as illustrated by the following example.

Example 3 *If* $j(p) = 1\text{-}max(p, p^2, \sin(p))$*, the constraint* $j(p) \leq j^+$ *can be expanded into the following primitive constraints:*

$$j(p) \leq j^+ \Leftrightarrow \left\{ \begin{array}{lll} & z_1 = p^2 & \\ and & z_2 = \sin(p) & \\ and & \left\{ \begin{array}{ll} or & \left\{ \begin{array}{ll} & z_3 = \max(p, z_1) \\ and & z_3 \in]-\infty, j^+] \end{array} \right. \\ & \left\{ \begin{array}{ll} & z_4 = \max(p, z_2) \\ and & z_4 \in]-\infty, j^+] \end{array} \right. \\ or & \left\{ \begin{array}{ll} & z_5 = \max(z_1, z_2) \\ and & z_5 \in]-\infty, j^+] \end{array} \right. \end{array} \right. \end{array} \right. \tag{18}$$

Because of the presence of the Boolean operator or*, the constraint is not* and*-decomposable and classical ICP cannot be applied.* ■

An adaptation of ICP to such *and-or*-decomposable problems will be made possible by the introduction of the new notion of set polynomials in Section 5. To implement RME, we also need a reliable procedure for global optimization. We shall use a trivial adaptation of a classical optimization algorithm [25], [9] presented in the following section. The resulting algorithm is particularly well suited to the use of ICP.

4 MINIMIZE algorithm

The problem to be solved is the minimization of a cost function $j(\mathbf{p})$ over a box $[\mathbf{p}_0]$. Let $\widehat{\mathcal{S}}$ be the set of all global minimizers of j over $[\mathbf{p}_0]$ and $[j]([\mathbf{p}])$ be an inclusion function for the cost function $j(\mathbf{p})$. The algorithm MINIMIZE, presented on Table 1, generates a list of boxes $\mathcal{S}^+$, the union of which contains $\widehat{\mathcal{S}}$, and computes an enclosure of the minimum $\widehat{j}$. It uses a local minimization procedure GODOWN (at Step 4), similar to that presented in [12], to decrease the upper bound j^+ for the minimum $\hat{j}$. Interval analysis is involved at Steps 10 and 11 to compute an inclusion function $[j]([\mathbf{p}])$ in order to enclose the range of j over the box $[\mathbf{p}]$. By taking advantage of the availability of the upper bound j^+, ICP is involved at Step 5 to replace $[\mathbf{p}]$ by a smaller box $[\mathbf{r}]$ such that any global minimizer in $[\mathbf{p}]$ is also in $[\mathbf{r}]$. $\mathcal{Q}$ is a First-In-First-Out list of boxes containing the part of the search space that has not yet been studied. Any point of $\mathcal{Q}$ is still a potential candidate for being a global minimizer. The real number $\varepsilon > 0$ is the width below which boxes will not be bisected. $[\widehat{j}]$ is an interval that contains the global minimum $\hat{j}$. It is computed by interval evaluation of j over all boxes of $\mathcal{S}^+$ at Step 11. The lower bound of the interval $[j]([\mathbf{p}]$ is denoted by $\mathrm{lb}([j]([\mathbf{p}]))$.

In the context of RME, two other problems remain to be solved, namely getting an inclusion function for $j_q(\mathbf{p})$ as defined by (1) and adapting an ICP-based procedure to the contraction of a box $[\mathbf{p}]$ under the constraint $j_q(\mathbf{p}) \leq j^+$, as required by Step 5 of MINIMIZE. These two

	MINIMIZE(in: $[\mathbf{p}_0]$, $j(.)$; out: $\widehat{[j]}$, $\mathcal{S}^+$)
1	$Q = \{[\mathbf{p}_0]\}$; $\mathcal{S}^+ = \emptyset$; $j^+ = \infty$; $\widehat{[j]} = \emptyset$;
2	while $Q \neq \emptyset$,
3	put first element of Q into $[\mathbf{p}]$;
4	$j^+ = \text{GODOWN}(\text{Center}([\mathbf{p}]), j(.))$;
5	find a box $[\mathbf{r}]$ such that $[\mathbf{p}] \cap j^{-1}(]-\infty, j^+]) \subset [\mathbf{r}] \subset [\mathbf{p}]$;
6	if $[\mathbf{r}] = \emptyset$, then go to 2;
7	if $(\text{width}([\mathbf{r}]) \leq \varepsilon)$, then $\{\mathcal{S}^+ = \mathcal{S}^+ \cup \{[\mathbf{r}]\}$; go to 2;$\}$
8	bisect $[\mathbf{r}]$ and append the two resulting boxes to Q;
9	end while
10	remove all $[\mathbf{p}]$ in $\mathcal{S}^+$ such that $\text{lb}([j]([\mathbf{p}])) > j^+$;
11	for all $[\mathbf{p}]$ in $\mathcal{S}^+$, $\widehat{[j]} = \widehat{[j]} \cup [j]([\mathbf{p}])$;
12	$\widehat{[j]} = \widehat{[j]} \cap]-\infty, j^+]$.

Table 1: Classical ICP-based algorithm for reliable minimization

operations will be based upon of a new type of object, namely *set polynomials*, presented in the next section.

5 Set polynomials

5.1 Introduction

The problem considered here is the contraction of a Cartesian domain $\mathcal{P}$ associated with the vector $\mathbf{p}$ subject to the constraint $j(\mathbf{p}) \in \mathcal{Y}$ when this constraint is *and*-*or*-decomposable. This problem has to be solved to implement Step 5 of MINIMIZE where the constraint to be taken into account is $j_q(\mathbf{p}) \in]-\infty, j^+]$. The notions developed in this section will also be used to obtain an inclusion function for the cost function $j_q(\mathbf{p})$, as needed at Steps 10 and 11 of MINIMIZE.

The principle of the procedure followed to contract $\mathcal{P}$ consists of three steps.

- **Step 1**: Decompose the constraint $j(\mathbf{p}) \in \mathcal{Y}$ into a set of m *and*-decomposable constraints related by the Boolean operator *or*. For instance, the constraint considered on Example 3 is decomposed as follows:

$$1\text{-}\max(p, p^2, \sin(p)) \leq j^+ \Leftrightarrow \left\{ \begin{array}{ll} & \max(p, p^2) \leq j^+ \\ \text{or} & \max(p^2, \sin(p)) \leq j^+ \\ \text{or} & \max(p, \sin(p)) \leq j^+. \end{array} \right. \tag{19}$$

- **Step 2**: Contract $\mathcal{P}$ with respect to each of the m constraints taken independently. Thus m Cartesian domains $\mathcal{P}(1), \ldots, \mathcal{P}(m)$ are obtained.

- **Step 3**: Compute $\mathcal{P}(1) \cup \cdots \cup \mathcal{P}(m)$.

The disjunctive decomposition at Step 1 leads to a combinatorial explosion in our context. It can be avoided by adapting this scheme: we shall use a specific decomposition to replace the disjunctive form at Step 1 and a specific set algorithm to replace the unions at Step 3. The theoretical background needed to understand this adaptation is presented in Sections 5.2 and 5.3, via the notion of set polynomials. In Section 5.4, a new procedure is given for contracting $\mathcal{P}$ under the constraint $j_q(\mathbf{p}) \in]-\infty, j^+]$ while avoiding combinatorial explosion.

5.2 Set polynomials

5.2.1 Definitions

The set function $\mathcal{F} : \mathcal{P}(\mathbb{R}^n) \times \cdots \times \mathcal{P}(\mathbb{R}^n) \to \mathcal{P}(\mathbb{R}^n)$, where $\mathcal{P}(\mathbb{R}^n)$ is the set of all subsets of $\mathbb{R}^n$, is *inclusion monotonic* if

$$\mathcal{X}(1) \subset \mathcal{Y}(1), \ldots, \mathcal{X}(m) \subset \mathcal{Y}(m) \Rightarrow \mathcal{F}(\mathcal{X}(1), \ldots, \mathcal{X}(m)) \subset \mathcal{F}(\mathcal{Y}(1), \ldots, \mathcal{Y}(m)). \quad (20)$$

For instance, $\mathcal{F}(\mathcal{X}(1), \mathcal{X}(2), \mathcal{X}(3)) \triangleq (\mathcal{X}(1) \cap \mathcal{X}(2)) \cup (\mathcal{X}(1) \cap \mathcal{X}(3))$ is inclusion monotonic, contrary to $\mathcal{F}(\mathcal{X}) \triangleq \bar{\mathcal{X}}$, where $\bar{\mathcal{X}}$ is the complementary set of $\mathcal{X}$ in $\mathbb{R}^n$.

Let $\mathcal{X}(i) \in \mathcal{P}(\mathbb{R}^n), i = 1, \ldots, m$, be m set indeterminates. The construction of the set $\mathcal{B}[\mathcal{X}(1), \ldots, \mathcal{X}(m)]$ of all polynomials in these indeterminates with coefficients in the set of Boolean numbers $\mathcal{B} \triangleq \{0, 1\}$ is, in principle, illicit, because $(\mathcal{P}(\mathbb{R}^n), \cup, \cap)$ is not a ring but only a semi-ring, since $(\mathcal{P}(\mathbb{R}^n), \cup)$ is only a monoid and not a group (see [11], page 116). By an abuse of notation commonly committed, *e.g.*, in the (max,+) community [1], we shall nevertheless speak of $\mathcal{B}[\mathcal{X}(1), \ldots, \mathcal{X}(m)]$ as a set of polynomials. One should keep in mind, however, that some classical operations allowed for ring-based polynomials are no longer valid. Any element of $\mathcal{B}[\mathcal{X}(1), \ldots, \mathcal{X}(m)]$ will be called a *set polynomial*. An example of a set polynomial is $(\mathcal{X}(1) \cap \mathcal{X}(2)) \cup (\mathcal{X}(1) \cap \mathcal{X}(1)) \cup \mathcal{X}(2)$, which is an element of $\mathcal{B}[\mathcal{X}(1), \mathcal{X}(2)]$. When there is no ambiguity, $\mathcal{A} \cup \mathcal{B}$ and $\mathcal{A} \cap \mathcal{B}$ will be denoted more concisely by $\mathcal{A} + \mathcal{B}$ and $\mathcal{A}.\mathcal{B}$, respectively. Set polynomials are obviously inclusion monotonic.

To enclose a set $\mathcal{S} = \mathcal{F}(\mathcal{X}(1), \ldots, \mathcal{X}(m))$ where $\mathcal{F}$ is inclusion monotonic, it suffices to enclose each $\mathcal{X}(k)$ into a box $\mathcal{Y}(k)$ (or more generally a Cartesian domain) and then to compute $\mathcal{F}(\mathcal{Y}(1), \ldots, \mathcal{Y}(m))$. A method for evaluating $\mathcal{F}(\mathcal{Y}(1), \ldots, \mathcal{Y}(m))$ is proposed in the following section.

5.2.2 Evaluation of set polynomials over Cartesian domains

Theorem 1 *(Cartesian expansion): Let $\mathcal{X}$ and $\mathcal{Y}$ be two Cartesian domains. Then*

$$\begin{array}{lrcl} (i) & (\mathcal{X}_1 \times \cdots \times \mathcal{X}_n) \cap (\mathcal{Y}_1 \times \cdots \times \mathcal{Y}_n) & = & (\mathcal{X}_1 \cap \mathcal{Y}_1) \times \cdots \times (\mathcal{X}_n \cap \mathcal{Y}_n) \\ (ii) & (\mathcal{X}_1 \times \cdots \times \mathcal{X}_n) \cup (\mathcal{Y}_1 \times \cdots \times \mathcal{Y}_n) & \subset & (\mathcal{X}_1 \cup \mathcal{Y}_1) \times \cdots \times (\mathcal{X}_n \cup \mathcal{Y}_n) \end{array} \tag{21}$$

Proof: The proof for (i) is trivial, and we shall only give a proof for (ii). First, recall that if $a_1, \ldots, a_n, b_1, \ldots, b_n$ are Boolean numbers, then,

$$\prod_{i=1}^{n} (a_i + b_i) = (a_1 a_2 \ldots a_{n-1} a_n) + (a_1 a_2 \ldots a_{n-1} b_n) + \cdots + (b_1 b_2 \ldots b_{n-1} b_n) \,. \tag{22}$$

Now, since in Boolean algebra, $a + b \geq a$, (22) implies that

$$\prod_{i=1}^{n} (a_i + b_i) \geq (a_1 a_2 \ldots a_{n-1} a_n) + (b_1 b_2 \ldots b_{n-1} b_n) \,, \tag{23}$$

or equivalently that [3]:

$$(a_1 a_2 \ldots a_{n-1} a_n) + (b_1 b_2 \ldots b_{n-1} b_n) \Rightarrow \prod_{i=1}^{n} (a_i + b_i) \,. \tag{24}$$

Let $\mathbf{z}$ be a vector in $\mathbb{R}^n$,

$$\begin{array}{ll} & \mathbf{z} \in (\mathcal{X}_1 \times \cdots \times \mathcal{X}_n) \cup (\mathcal{Y}_1 \times \cdots \times \mathcal{Y}_n) \\ \Leftrightarrow & ((z_1 \in \mathcal{X}_1) \text{ and } \ldots \text{ and } (z_n \in \mathcal{X}_n)) \text{ or } ((z_1 \in \mathcal{Y}_1) \text{ and } \ldots \text{ and } (z_n \in \mathcal{Y}_n)) \\ \overset{(24)}{\Rightarrow} & ((z_1 \in \mathcal{X}_1) \text{ or } (z_1 \in \mathcal{Y}_1)) \text{ and } \ldots \text{ and } ((z_n \in \mathcal{X}_n) \text{ or } (z_n \in \mathcal{Y}_n)) \\ \Leftrightarrow & \mathbf{z} \in (\mathcal{X}_1 \cup \mathcal{Y}_1) \times \cdots \times (\mathcal{X}_n \cup \mathcal{Y}_n) \,. \end{array} \tag{25}$$

■

Theorem 2 *(*Cartesian decomposition*): If $\mathcal{X}(1), \ldots, \mathcal{X}(m)$ are m Cartesian domains of $\mathbb{R}^n$, and $\mathcal{F}$ is a set polynomial, then*

$$\mathcal{F}(\mathcal{X}(1), \ldots, \mathcal{X}(m)) \subset \mathcal{F}(\mathcal{X}_1(1), \ldots, \mathcal{X}_n(1)) \times \cdots \times \mathcal{F}(\mathcal{X}_1(m), \ldots, \mathcal{X}_n(m)) \,. \tag{26}$$

This theorem is a direct consequence of Theorem 1. We shall only give a sketch its proof on an academic example.

Example 4 *: Assume that*

$$\mathcal{F}(\mathcal{X}, \mathcal{Y}, \mathcal{Z}) = \mathcal{X}.\mathcal{Y} + \mathcal{Y}.\mathcal{Z}, \tag{27}$$

[3] Recall that, in the set of Boolean number $\{0, 1\}$, $a \geq b$ implies that if $b = 1$, then $a = 1$, *i.e.*, if b is true, a is also true, which can be written as $b \Rightarrow a$.

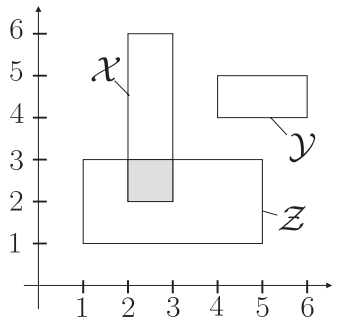

Figure 2: The grey box is the value of the set polynomial $\mathcal{F}(\mathcal{X},\mathcal{Y},\mathcal{Z})$ defined by (29).

with $n=2$. *Then*

$$\begin{array}{rcl}\mathcal{F}(\mathcal{X},\mathcal{Y},\mathcal{Z}) & \overset{(27)}{=} & (\mathcal{X}_1\times\mathcal{X}_2).(\mathcal{Y}_1\times\mathcal{Y}_2)+(\mathcal{Y}_1\times\mathcal{Y}_2).(\mathcal{Z}_1\times\mathcal{Z}_2)\\ & \overset{(21,\,i)}{=} & ((\mathcal{X}_1.\mathcal{Y}_1)\times(\mathcal{X}_2.\mathcal{Y}_2))+((\mathcal{Y}_1.\mathcal{Z}_1)\times(\mathcal{Y}_2.\mathcal{Z}_2))\\ & \overset{(21,\,ii)}{\subset} & (\mathcal{X}_1.\mathcal{Y}_1+\mathcal{Y}_1.\mathcal{Z}_1)\times(\mathcal{X}_2.\mathcal{Y}_2+\mathcal{Y}_2.\mathcal{Z}_2)\\ & = & \mathcal{F}(\mathcal{X}_1,\mathcal{Y}_1,\mathcal{Z}_1)\times\mathcal{F}(\mathcal{X}_2,\mathcal{Y}_2,\mathcal{Z}_2)\,. \qquad \blacksquare\end{array} \tag{28}$$

When $\mathcal{X}(1),\ldots,\mathcal{X}(m)$ are Cartesian domains and $\mathcal{F}$ is a set polynomial, the smallest Cartesian domain containing $\mathcal{F}(\mathcal{X}(1),\ldots,\mathcal{X}(m))$ can be computed exactly by rewriting $\mathcal{F}$ in disjunctive form. Consider, for instance, the situation of Figure 2, with

$$\mathcal{F}(\mathcal{X},\mathcal{Y},\mathcal{Z})=(\mathcal{X}\cap\mathcal{Y})\cup(\mathcal{Y}\cap\mathcal{Z})\cup(\mathcal{X}\cap\mathcal{Z})\,. \tag{29}$$

$\mathcal{F}(\mathcal{X},\mathcal{Y},\mathcal{Z})$ is the box painted grey:

$$\mathcal{F}(\mathcal{X},\mathcal{Y},\mathcal{Z})=\emptyset\cup\emptyset\cup(\mathcal{X}\cap\mathcal{Z})=[2,3]\times[2,3]. \tag{30}$$

However, an exact procedure to evaluate it via the computation of a disjunctive form may become too complex when the number m of sets increases, because this disjunctive form is usually longer than the initial form. To avoid this, one may use Cartesian decomposition. Since (29) is already in disjunctive form (a very special case), the complexity of using the Cartesian decomposition remains the same, but the result is now pessimistic:

$$\begin{array}{rl} & \mathcal{F}(\mathcal{X}_1,\mathcal{Y}_1,\mathcal{Z}_1)\times\mathcal{F}(\mathcal{X}_2,\mathcal{Y}_2,\mathcal{Z}_2)\\ = & ((\mathcal{X}_1\cap\mathcal{Y}_1)\cup(\mathcal{Y}_1\cap\mathcal{Z}_1)\cup(\mathcal{X}_1\cap\mathcal{Z}_1))\times((\mathcal{X}_2\cap\mathcal{Y}_2)\cup(\mathcal{Y}_2\cap\mathcal{Z}_2)\cup(\mathcal{X}_2\cap\mathcal{Z}_2))\\ = & ([2,3]\cap[4,6]\cup[4,6]\cap[1,5]\cup[2,3]\cap[1,5])\times([2,6]\cap[4,5]\cup[4,5]\cap[1,3]\cup[2,6]\cap[1,3])\\ = & (\emptyset\cup[4,5]\cup[2,3])\times([4,5]\cup\emptyset\cup[2,3])=([2,3]\cup[4,5])\times([2,3]\cup[4,5])\,.\end{array}$$

The resulting Cartesian domain thus consists of four boxes, and is an outer approximation of the actual solution $\mathcal{F}(\mathcal{X},\mathcal{Y},\mathcal{Z})=[2,3]\times[2,3]$. $\blacksquare$

5.3 Case of elementary set polynomials

We now focus our attention on constraints of the form $j_q(\mathbf{p}) \leq j^+$, as needed by Step 5 of MINIMIZE. We first recall the definition of symmetric set polynomials and show how they govern the relations between the *and*-decomposable constraints of the robust minimax estimation problem. Then, we propose a recursive definition of symmetric polynomials, in order to derive an efficient algorithm for their outer evaluation over Cartesian domains.

5.3.1 Definitions

A set polynomial $\mathcal{F}(\mathcal{X}(1), \ldots, \mathcal{X}(m))$ is *symmetric* if it is invariant under permutation. For instance, $\mathcal{X}(1)\mathcal{X}(2) + \mathcal{X}(1)\mathcal{X}(3) + \mathcal{X}(2)\mathcal{X}(3) + \mathcal{X}(1)\mathcal{X}(2)\mathcal{X}(3)$ is symmetric. The *elementary symmetric set polynomials* are defined (by analogy with [11], page 133) as

$$\Phi_0(m) = \prod_{i=1}^{m} \mathcal{X}(i), \quad \Phi_1(m) = \prod_{i<j}^{m} \mathcal{X}(i) + \mathcal{X}(j), \quad \Phi_2(m) = \prod_{i<j<k}^{m} \mathcal{X}(i) + \mathcal{X}(j) + \mathcal{X}(k), \ldots \quad (31)$$

$$\Phi_{m-3}(m) = \sum_{i<j<k}^{m} \mathcal{X}(i)\mathcal{X}(j)\mathcal{X}(k), \quad \Phi_{m-2}(m) = \sum_{i<j}^{m} \mathcal{X}(i)\mathcal{X}(j), \quad \Phi_{m-1}(m) = \sum_{i=1}^{m} \mathcal{X}(i). \quad (32)$$

By convention, $\Phi_m(m) = \mathbb{R}^n$. We shall call $\Phi_q(m)$ the *q-intersection* of the m sets $\mathcal{X}(k), k \in \{1, \ldots, m\}$, as it is the set of all $\mathbf{x}$'s that belong to at least $m - q$ of these sets. Since

$$j_q(\mathbf{p}) \leq j^+ \Leftrightarrow \left\{ \begin{array}{l} \mathbf{p} \in \Phi_q(m) \\ \text{with } \mathcal{X}_i = f_i^{-1}(]-\infty, j^+]), i \in \{1, \ldots, m\} \end{array} \right. , \quad (33)$$

an enclosure of $\mathcal{S}_q(j^+) \triangleq \{\mathbf{p} \in [\mathbf{p}] \mid j_q(\mathbf{p}) \leq j^+\}$ could be obtained by expanding Φ_q into its disjunctive form. Unfortunately, this expansion gives rise to a combinatorial explosion. For instance, if $m = 10$ and $q = 4$, which corresponds to 10 measurements with at most 4 outliers, Φ_q is the sum of 210 monomials. This combinatorial explosion can be avoided using a recursive definition for Φ_q, presented next.

5.3.2 Recursive definition of the elementary set symmetric polynomials

The next theorem provides a new way to evaluate $\Phi_r(m), 0 \leq r \leq q$, efficiently and recursively over m.

Theorem 3 *: Assume that* $\Phi_0(m-1),\ldots,\Phi_q(m-1)$ *are available and that a new set* $\mathcal{X}(m)$ *has to be taken into account. Then* $\Phi_0(m),\ldots,\Phi_q(m)$ *can be obtained recursively as follows:*

$$\begin{bmatrix} \Phi_0(m) \\ \Phi_1(m) \\ \vdots \\ \Phi_q(m) \end{bmatrix} = \begin{bmatrix} \Phi_0(m-1)\,\mathcal{X}(m) \\ \Phi_1(m-1)\,\mathcal{X}(m) \cup \Phi_0(m-1) \\ \vdots \\ \Phi_q(m-1)\,\mathcal{X}(m) \cup \Phi_{q-1}(m-1) \end{bmatrix} \tag{34}$$

where $\Phi_k(0) = \mathbb{R}^n$ *and* $\Phi_k(k) = \mathbb{R}^n$ *if* $k \in \{0,\ldots,q\}$. ■

In a $(\cup,\cap)$-algebra, this recursive equation can thus be interpreted as a nonlinear discrete-time set system where the state vector is $\Phi(m) = [\Phi_0(m)\ \ldots\ \Phi_q(m)]^{\mathrm{T}}$ and the input is $\mathcal{X}(m)$.

Proof: The proof is a direct application of Horner's scheme. To avoid introducing new notations and tedious manipulations of indices, we shall restrict ourselves to checking (34) for $m = 4$.

$$\begin{bmatrix} \Phi_0(4) \\ \Phi_1(4) \\ \Phi_2(4) \\ \Phi_3(4) \end{bmatrix} \triangleq \begin{bmatrix} \mathcal{X}_1\mathcal{X}_2\mathcal{X}_3\mathcal{X}_4 \\ \mathcal{X}_1\mathcal{X}_2\mathcal{X}_3 + \mathcal{X}_1\mathcal{X}_2\mathcal{X}_4 + \mathcal{X}_1\mathcal{X}_3\mathcal{X}_4 + \mathcal{X}_2\mathcal{X}_3\mathcal{X}_4 \\ \mathcal{X}_1\mathcal{X}_2 + \mathcal{X}_1\mathcal{X}_3 + \mathcal{X}_1\mathcal{X}_4 + \mathcal{X}_2\mathcal{X}_3 + \mathcal{X}_2\mathcal{X}_4 + \mathcal{X}_3\mathcal{X}_4 \\ \mathcal{X}_1 + \mathcal{X}_2 + \mathcal{X}_3 + \mathcal{X}_4 \end{bmatrix}$$

$$= \begin{bmatrix} (\mathcal{X}_1\mathcal{X}_2\mathcal{X}_3)\,\mathcal{X}_4 \\ (\mathcal{X}_1\mathcal{X}_2 + \mathcal{X}_1\mathcal{X}_3 + \mathcal{X}_2\mathcal{X}_3)\,\mathcal{X}_4 + (\mathcal{X}_1\mathcal{X}_2\mathcal{X}_3) \\ (\mathcal{X}_1 + \mathcal{X}_2 + \mathcal{X}_3)\,\mathcal{X}_4 + (\mathcal{X}_1\mathcal{X}_2 + \mathcal{X}_1\mathcal{X}_3 + \mathcal{X}_2\mathcal{X}_3) \\ \mathbb{R}^n\mathcal{X}_4 + (\mathcal{X}_1 + \mathcal{X}_2 + \mathcal{X}_3) \end{bmatrix} = \begin{bmatrix} \Phi_0(3)\,\mathcal{X}(4) \\ \Phi_1(3)\,\mathcal{X}(4) + \Phi_0(3) \\ \Phi_2(3)\,\mathcal{X}(4) + \Phi_1(3) \\ \Phi_3(3)\,\mathcal{X}(4) + \Phi_2(3) \end{bmatrix}$$

■

The following theorem gives three basic properties of the set polynomials $\Phi_q(m)$.

Theorem 4 *The following properties hold true.*

$$\begin{aligned} &\textit{(i)} && \Phi_0(m) \subset \Phi_1(m) \subset \cdots \subset \Phi_q(m), \\ &\textit{(ii)} && \Phi_q(m) \subset \Phi_q(m-1) \subset \cdots \subset \Phi_q(q) = \cdots = \Phi_q(0) = \mathbb{R}^n, \\ &\textit{(iii)} && \mathcal{X}(k) \cap \Phi_q(m) = \emptyset \Rightarrow \Phi_q(m) = \Phi_{q-1}(\mathcal{X}(1),\ldots,\mathcal{X}(k-1),\mathcal{X}(k+1),\ldots,\mathcal{X}(m)). \end{aligned} \tag{35}$$

Proof: (i) $\Phi_k(m)$ is the sum of all monomials of the form $\Sigma_k = \mathcal{X}(i_1)\mathcal{X}(i_2)\mathcal{X}(i_3)\ldots\mathcal{X}(i_{m-k})$. Now, the monomial $\Sigma_{k+1} = \mathcal{X}(i_1)\mathcal{X}(i_2)\mathcal{X}(i_3)\ldots\mathcal{X}(i_{m-k-1})$ is a monomial of $\Phi_{k+1}(m)$. Thus, any monomial of $\Phi_k(m)$ is included in $\Phi_{k+1}(m)$. Therefore $\Phi_k(m) \subset \Phi_{k+1}(m)$.
(ii) $\Phi_q(m) = \Phi_q(m-1)\,\mathcal{X}(m) \cup \Phi_{q-1}(m-1)$. Now, from (i), $\Phi_{q-1}(m-1) \subset \Phi_q(m-1)$ so $\Phi_q(m) \subset \Phi_q(m-1)\,\mathcal{X}(m) \cup \Phi_q(m-1) = \Phi_q(m-1)$.
(iii) Set $\Phi_q^k(m) = \Phi_q(\mathcal{X}(1),\ldots,\mathcal{X}(k-1),\mathcal{X}(k+1),\ldots,\mathcal{X}(m))$. Factor $\Phi_q(m)$ with respect

	q-INTERSECTION(in: $\mathcal{X}(1), \ldots, \mathcal{X}(m)$; out: $[\Phi_q(m)]$)
1	For $k \in \{0, \ldots, m\}$, $[\Phi_{-1}](k) = \emptyset$;
2	For $\ell \in \{0, \ldots, q\}$, $[\Phi_\ell](0) = \mathbb{R}^n$;
3	For $k = 1$ up to m
4	For $\ell = 0$ up to q
5	$[\Phi_\ell(k)] = ([\Phi_\ell](k-1) \cap \mathcal{X}(k)) \cup [\Phi_{\ell-1}](k-1)$;

Table 2: *Evaluation of an enclosure of the set polynomial Φ_q over m Cartesian domains*

to $\mathcal{X}(k)$ to get $\Phi_q(m) = \Phi_q^k(m)\mathcal{X}(k) \cup \Phi_{q-1}^k(m)$. Intersect both sides of this equation with $\Phi_q(m)$ to get $\Phi_q(m) = \Phi_q^k(m)\mathcal{X}(k)\,\Phi_q(m) \cup \Phi_{q-1}^k(m)\Phi_q(m)$. Now, by assumption, $\mathcal{X}(k) \cap \Phi_q(m) = \emptyset$. Therefore $\Phi_q(m) = \emptyset \cap \Phi_{q-1}^k(m)\Phi_q(m)$ which implies that $\Phi_q(m) = \Phi_{q-1}^k(m)$. ■

In the context of bounded-error estimation, Theorem 4 can be interpreted as follows: $\mathbf{p} \in \mathcal{X}(k)$ means that $\mathbf{p}$ is consistent with the kth datum; (i) if $\mathbf{p}$ is consistent with at least $m-q$ data, then it is also consistent with at least $m-q-1$ of them; (ii) if $\mathbf{p}$ is consistent with at least $m-q$ of the m first data, then it is also consistent with at least $m-1-q$ of the $m-1$ first data; (iii) if there exists no $\mathbf{p}$ consistent with at least $m-q$ of the data and also with the kth datum, then one can remove the kth datum (which is interpreted as an outlier) from the data set and replace q by $q-1$ to get a simpler definition of $\Phi_q(m)$.

5.3.3 Evaluation over Cartesian domains

To implement the recursive procedure (34), one should use sets on which unions or intersections can be computed or at least enclosed, such as Cartesian domains, boxes, ellipsoids or polytopes. Unfortunately, the enclosure with such sets introduces pessimism and the equality (34) becomes an inclusion. The q-INTERSECTION algorithm given in Table 2 computes a Cartesian domain that encloses the set $\Phi_\ell(\mathcal{X}(1), \ldots, \mathcal{X}(m))$, $\ell \in \{1, \ldots, q\}$, where the sets $\mathcal{X}(k)$ are Cartesian domains. In the computer, $[\Phi_\ell](k)$, $\ell \in \{-1, \ldots, q\}$, $k \in \{0, \ldots, m\}$ is represented as a $((q+2) \times (m+1))$-matrix, the entries of which are Cartesian domains.

This algorithm will now be illustrated on the situation described by Figure 3, where the Cartesian domains to be considered are boxes. For $q = 0, 1, 2$, q-INTERSECTION respectively yields $[\Phi_0]([\mathbf{x}](1), \ldots, [\mathbf{x}](9)) = \emptyset$, $[\Phi_1]([\mathbf{x}](1), \ldots, [\mathbf{x}](9)) = \emptyset$ and $[\Phi_2]([\mathbf{x}](1), \ldots, [\mathbf{x}](9)) \subset [6,7] \times [5,6]$ (represented by the black box in Figure 2). The results obtained here are exact, but in general they are pessimistic. To limit such a pessimism, a possible improvement based Theorem 4 (iii) can be used. This improvement has not been implemented yet.

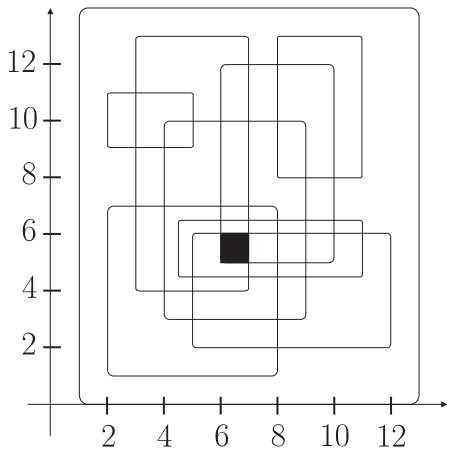

Figure 3: The black box is the 2-intersection of 9 boxes.

5.4 Application to robust minimax estimation

In this paragraph, we show how set polynomials can be used to contract $[\mathbf{p}]$ under the constraint $j_q(\mathbf{p}) \leq j^+$, as required by Step 5 of MINIMIZE in the context of robust minimax estimation, and to obtain an inclusion function for j_q, as required by Steps 10 and 11. The contraction of $[\mathbf{p}]$ is a direct application of the fact that

$$[\mathbf{p}] \cap j_q^{-1}(]-\infty, j^+]) = \Phi_q\left([\mathbf{p}] \cap f_1^{-1}(]-\infty, j^+], \ldots, [\mathbf{p}] \cap f_m^{-1}(]-\infty, j^+])\right), \tag{36}$$

(see (33)). The following algorithm contracts $[\mathbf{p}]$.

CONTRACT(in: $[\mathbf{p}]$, j^+; out: $[\mathbf{r}]$);
1 for $k = 1$ to m, compute a Cartesian domain $\mathcal{P}(k)$
such that $\left([\mathbf{p}] \cap f_k^{-1}(]-\infty, j^+])\right) \subset \mathcal{P}(k) \subset [\mathbf{p}]$;
2 $\mathcal{R} = q$-INTERSECTION$(\mathcal{P}(1), \ldots, \mathcal{P}(m))$;
3 return $[\mathbf{r}]$, the smallest box that contains $[\mathbf{p}] \cap \mathcal{R}$.

Since the contraint $f_k(\mathbf{p}) \leq j^+$, is *and*-decomposable, $\mathcal{P}(k)$ can be obtained by using interval constraint propagation as explained in Section 3.2.

To get an inclusion function for $j_q(\mathbf{p})$, consider the semi-ring $(\mathbb{R}, \min, \max)$. Denote the qth elementary symmetric polynomial in m indeterminates by $\Phi_q(x_1, \ldots, x_m)$. For instance,

$$\begin{aligned}
\Phi_0(x_1, x_2, x_3) &= \max(x_1, x_2, x_3) \\
\Phi_1(x_1, x_2, x_3) &= \min(\max(x_1, x_2), \max(x_2, x_3), \max(x_1, x_3)) \\
\Phi_2(x_1, x_2, x_3) &= \min(x_1, x_2, x_3).
\end{aligned} \tag{37}$$

The cost function $j_q(\mathbf{p})$ of (1) can then be expressed as

$$j_q(\mathbf{p}) = j_q(m, \mathbf{p}) = \Phi_q(f_1(\mathbf{p}), \ldots, f_m(\mathbf{p})). \tag{38}$$

and the recursive implementation (34) can be adapted to this context to compute $j_q(\mathbf{p})$:

$$\begin{bmatrix} j_0\,(m,\mathbf{p}) \\ j_1\,(m,\mathbf{p}) \\ \vdots \\ j_q\,(m,\mathbf{p}) \end{bmatrix} = \begin{bmatrix} \max(j_0\,(m-1,\mathbf{p})\,, f_m(\mathbf{p})) \\ \min(\max(j_1\,(m-1,\mathbf{p})\,, f_m(\mathbf{p})), j_0\,(m-1,\mathbf{p})) \\ \vdots \\ \min(\max(j_q\,(m-1,\mathbf{p})\,, f_m(\mathbf{p})), j_{q-1}\,(m-1,\mathbf{p})) \end{bmatrix} \tag{39}$$

where $j_\ell(k,\mathbf{p}) = \Phi_\ell(f_1(\mathbf{p}),\ldots,f_k(\mathbf{p}))$; $k \in 1,\ldots,m$; $k > q$; $j_k(0,\mathbf{p}) = -\infty$ and $j_k(k,\mathbf{p}) = -\infty$. An inclusion function for $j_q(\mathbf{p})$, can thus be derived from (39) by applying the rules of interval computation and by returning the interval enclosure of $j_q(m,[\mathbf{p}])$.

6 Test case

Consider a two-exponential model where the relation between the parameter vector $\mathbf{p}$ and the model output is given by

$$y_{\mathrm{m}}(\mathbf{p},t) = p_1 \exp(-p_2 t) + p_3 \exp(-p_4 t). \tag{40}$$

Since a permutation of p_1 with p_3 and of p_2 with p_4 does not affect the model output y_{m}, the model is not globally identifiable. Therefore, any reliable identification method should lead to symmetrical solutions, if the search domain is large enough. Assume that ten data points $y(1),\ldots,y(10)$ are generated as follows. First, a noise-free data vector $\mathbf{y}^*$ is computed. Its 10 components are obtained by computing $y_{\mathrm{m}}(\mathbf{p}^*,t_k)$ as given by (40) for $\mathbf{p}^* = (20, 0.8, -10, 0.2)^{\mathrm{T}}$ and $t_k = \frac{1}{4}k^2, k \in \{1,\ldots,10\}$. Noisy data are then obtained by adding to each component y_k^* of $\mathbf{y}^*$ the noise $n_k = (|y_k^*| + 5) * \eta_k$, where η_k is a random noise with a uniform distribution in the interval $[-0.1, 0.1]$. The resulting data vector $\mathbf{y}^0$ is the regular data vector. A second data vector $\mathbf{y}^1$ is obtained by replacing y_3 by 30 in $\mathbf{y}^0$, and a third one $\mathbf{y}^2$ by replacing y_8 by -30 in $\mathbf{y}^1$. For $\varepsilon = 0.05$ and $[\mathbf{p}_0] = [-40, 40] \times [0, 1] \times [-40, 40] \times [0, 1]$, the results obtained by using RME as described in Section 2 are summarized in Table 3, where $\#\mathcal{S}_q^+$ is the number of boxes of $\mathcal{S}_q^+$ and #split is the number of bisections performed by Minimize. All computing times are for a Pentium 133MHz. Note that for a given data vector $\mathbf{y}^i$, $j_q(\mathbf{p}^*)$ and $\hat{j}_q\,(\mathbf{p})$ are decreasing when q increases as expected. At each run, $\mathcal{S}_q^+$ turns out to consist of two connected components. One of them, denoted by $\mathcal{S}_q^+\,(1)$, belongs to the half space where $p_1 > 0$ and the second one $\mathcal{S}_q^+\,(2)$ belongs to the half space where $p_1 < 0$). The smallest boxes guaranteed to contain $\mathcal{S}_q^+\,(1)$ (*i.e.,* the set associated with $\mathbf{p}^*$) obtained are given in Table 4, which evidences the fact that reasonable estimates are obtained only provided that q is equal to or greater than the actual number of outliers q^*.

Various strategies can be thought to for the choice of q, the maximum number of tolerated outliers. A reasonable guide line is to iterate the minimization of $j_q(\mathbf{p})$ over $[\mathbf{p}_0]$ for $q = 0, 1, 2, \ldots$ until the results obtained lead one to believe that q is greater than the actual number of outliers

data set	q	#split	time (sec.)	$\#\mathcal{S}_q^+$	$[\hat{\jmath}_q]$	$j_q(\mathbf{p}^*)$
$\mathbf{y}^0$	0	323	20,7	42	$[0.4415, 0.4577]$	1.067
$\mathbf{y}^0$	1	845	69.6	18	$[0.3719, 0.3748]$	0.694
$\mathbf{y}^0$	2	6044	770.0	21	$[0.1983, 0.2138]$	0.465
$\mathbf{y}^1$	0	1581	83.8	230	$[14.056, 14.097]$	33.070
$\mathbf{y}^1$	1	341	27.8	70	$[0.4429, 0.4582]$	1.067
$\mathbf{y}^1$	2	906	104.0	10	$[0.3728, 0.3746]$	0.694
$\mathbf{y}^2$	0	298	15.16	70	$[17.605, 17.651]$	33.070
$\mathbf{y}^2$	1	2779	220.1	246	$[14.059, 14.097]$	29,59
$\mathbf{y}^2$	2	343	41.5	59	$[0.4418, 0.4580]$	1.067

Table 3: Results obtained by the robust minimax estimator for three data sets with 0, 1 and 2 outliers respectively and for three values of q (first part)

data set	q	Box guaranteed to contain $\mathcal{S}_q^+$ (1)
$\mathbf{y}^0$	0	$[18.31, 18.63] \times [0.9043, 0.9244] \times [-7.622, -7.405] \times [0.1607, 0.1648]$
$\mathbf{y}^0$	1	$[17.78, 17.93] \times [0.9952, 1] \times [-6.628, -6.558] \times [0.1582, 0.1590]$
$\mathbf{y}^0$	2	$[19.16, 19.31] \times [0.9924, 1] \times [-7.672, -7.573] \times [0.1891, 0.1919]$
$\mathbf{y}^1$	0	$[39.45, 40] \times [0.300, 0.3042] \times [-40, -39.45] \times [0.988, 1]$
$\mathbf{y}^1$	1	$[18.41, 18.64] \times [0.9045, 0.9252] \times [-7.602, -7.4767] \times [0.1620, 0.1644]$
$\mathbf{y}^1$	2	$[17.79, 18.17] \times [0.9928, 1] \times [-6.734, -6.5618] \times [0.1583, 0.1602]$
$\mathbf{y}^2$	0	$[39.942, 40] \times [0.1369, 0.1514] \times [-17.087, -16.15] \times [0.001, 0.0012]$
$\mathbf{y}^2$	1	$[39.398, 40] \times [0.300, 0.3042] \times [-40, -39.45] \times [0.9878, 1]$
$\mathbf{y}^2$	2	$[18.38, 18.64] \times [0.9035, 0.9275] \times [-7.6531, -7.455] \times [0.1617, 0.1656]$

Table 4: Results obtained by the robust minimax estimator for three data sets with 0, 1 and 2 outliers respectively and for three values of q (second part)

q^*. No systematic procedure exists for the detection of q^*. Nevertheless,as illustrated by the example in this section, if the optimizers are on the border of $[\mathbf{p}_0]$ or if the value of $\hat{j}_q$ is too large, we can suspect that there are at least $q+1$ outliers. Note that the dimension of $\hat{j}_q$ is that of the output error and one often knows some bound on the largest error one is prepared to accept. Moreover, if $\hat{j}_q - \hat{j}_{q+1}$ is large the reliability of the datum y_i which satisfies $\hat{j}_q = |y_i - y_{\mathrm{m},i}(\hat{\mathbf{p}})|$ for one $\hat{\mathbf{p}} \in \hat{\mathcal{S}}_q$ is questionable since its presence modify the estimation results heavily. These comments suggest the following strategy for the choice of q: take the smallest q such that, (i) $\hat{j}_q$ is acceptably small, (ii) $\hat{j}_q$ is close to $\hat{j}_{q+1}$, and (iii) $\hat{\mathcal{S}}_q$ is not on the border of the search box $[\mathbf{p}_0]$.

7 Conclusions

When the noise corrupting the data can be assumed to belong to a sequence of random variables that are independently uniformly distributed over the interval $[-\delta, \delta]$, with δ unknown, minimax estimation is a standard approach for the identification of the model parameters, because the resulting estimated parameter vector belongs to the set of all maximum-likelihood estimates for any value of δ such that this set is not empty. Minimax estimation is however seldom used in practice, because of its well known sensitivity to outliers. The procedure described in this paper makes a minimax robust estimator robust to a prespecified number of data points that can take arbitrary values. It does so in a guaranteed way, by enclosing the set of robust minimax estimates thus defined in a union of boxes in parameter space, even in the case where the model is nonlinear in its parameters and the estimates are not unique, as illustrated by an example. To the best of our knowledge, there is no other method available in the literature to deal with this type of problem.

There are many reasons why combinatorial complexity looms over any attempt at solving such a problem, and several measures were taken to limit it as much as possible. First, it should be stressed that the various combinations of up to q outliers among m data points are not considered in isolation but collectively. The other measures for limiting complexity while preserving guaranteedness result from the combination of three tools: *interval analysis* which provides guaranteed results on the computations, *constraint propagation* to efficiently eliminate large parts of the search space without requiring bisections, and *set polynomials* to deal with Boolean connections between the constraints.

The notion of *and-or*-decomposable function, motivated by robust minimax estimation and introduced in this paper, represents a significant extension of the class of problems to be considered with interval constraint propagation. As such, it opens up new possibilities of research in this relatively young field, which should also lead to improvements in the algorithms for guaranteed nonlinear estimation in the presence of outliers.

References

[1] F. Baccelli, G. Cohen, G.J. Olsder, and J.P. Quadrat. *Synchronization and Linearity. An Algebra for Discrete Event Systems.* John Wiley & Sons, New York, 1992.

[2] E.W. Bai and H. Cho. Minimization with few violated constraints and its application in set-membership identification. In *Proc. IFAC Word Congress*, volume H, pages 343–348, Beijing, China, 1999.

[3] F. Benhamou and L. Granvilliers. Automatic generation of numerical redundancies for nonlinear constraint solving. *Reliable Computing*, pages 335–344, 1997.

[4] F. Benhamou and W. Older. Applying interval arithmetic to real, integer and Boolean constraints. *Journal of Logic Programming*, pages 1–24, 1997.

[5] F. L. Chernousko. *State Estimation for Dynamic Systems.* CRC Press, 1994.

[6] J. C. Cleary. Logical arithmetic. *Future Computing Systems*, 2(2):125–149, 1987.

[7] E. Davis. Constraint propagation with interval labels. *Artificial Intelligence*, 32:281–331, 1987.

[8] E. R. Hansen. *Global Optimization using Interval Analysis.* Marcel Dekker, New York, 1992.

[9] P. Van Hentenryck, Y. Deville, and L. Michel. *Numerica. A Modeling Language for Global Optimization.* MIT Press, Boston, 1997.

[10] E. Hyvonen. Constraint reasoning based on interval arithmetic; the tolerance propagation approach. *Artificial Intelligence*, 58:71–112, 1992.

[11] N. Jacobson. *Basic Algebra.* Freeman and Company, San Francisco, 1974.

[12] L. Jaulin. Interval constraint propagation with application to bounded-error estimation. *Automatica (accepted for publication)*, 2000.

[13] L. Jaulin. Reliable minimax parameter estimation. *Reliable Computing (to appear)*, 2000.

[14] A. Kurzhanski and I. Valyi. *Ellipsoidal Calculus for Estimation and Control.* Birkhäuser, Boston, 1996.

[15] H. Lahanier, E. Walter, and R. Gomeni. OMNE: a new robust membership-set estimator for the parameters of nonlinear models. *J. of Pharmacokinetics and Biopharmaceutics*, 15:203–219, 1987.

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

[17] R. E. Moore. *Methods and Applications of Interval Analysis*. SIAM Publ., Philadelphia, 1979.

[18] J. P. Norton (Ed.). Special issue on bounded-error estimation: Issue 1. *Int. J. of Adaptive Control and Signal Processing*, 8(1):1–118, 1994.

[19] J. P. Norton (Ed.). Special issue on bounded-error estimation: Issue 2. *Int. J. of Adaptive Control and Signal Processing*, 9(1):1–132, 1995.

[20] P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley and Sons, New York, 1987.

[21] E. Walter (Ed.). Special issue on parameter identifications with error bounds. *Mathematics and Computers in Simulation*, 32(5&6):447–607, 1990.

[22] D.L. Waltz. Generating semantic descriptions from drawings of scenes with shadows. In P.H. Winston, editor, *The Psychology of Computer Vision*, pages 19–91, New-York, 1975. McGraw-Hill.

[23] M.A. Wolfe. Interval methods for global optimization. *Applied Mathematics and Computation*, 75:179–206, 1996.

[24] M.A. Wolfe. On discrete minimax problems in the set of real numbers using interval arithmetic. *Reliable Computing*, 5(4):371–383, 1999.

[25] J. Zhou. A permutation-based approach for solving the job-shop problem. *Constraints*, 1:1–30, 1996.