Guaranteed nonlinear estimation using constraint propagation on sets

Luc Jaulin [1] [2], Michel Kieffer[1] , Isabelle Braems[1] and Eric Walter[1]

Abstract: Bounded-error estimation is the estimation of the parameter or state vector of a model from experimental data, under the assumption that some suitably defined errors should belong to some prior feasible sets. When the model outputs are linear in the vector to be estimated, a number of methods are available to contain all estimates that are consistent with the data within simple sets such as ellipsoids, orthotopes or parallelotopes, thereby providing guaranteed set estimates. In the nonlinear case, the situation is much less developed and there are very few methods that produce such guaranteed estimates. In this paper, the problem of characterizing the set of all state vectors that are consistent with all data in the case of nonlinear discrete-time systems is cast into the more general framework of constraint satisfaction problems. The state vector at time k should be estimated either on-line from past measurement only or off-line from a series of measurements that may include measurements posterior to k. Even in the causal case, prior information on the future value of the state and output vectors, due for instance to physical constraints, is readily taken into account. Algorithms taken from the literature of interval constraint propagation are extended by replacing intervals by more general subsets of real vector spaces. This makes it possible to propose a new algorithm that contracts the feasible domains for each uncertain variable optimally (*i.e.*, no smaller domain could be obtained) and efficiently.

Keywords: bounded-error estimation, constraint propagation, CSP, identification, interval analysis, nonlinear estimation, observation, set estimation.

1 Introduction

In a linear context, many tools are available to estimate the parameter or state vector of a model from experimental data. They can be classified according to how they deal with uncertainty. Some of them do not take explicitly into account the fact that the model is an approximation of reality and that the measurements are corrupted by noise. This is the case, for instance, of

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

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

Luenberger state observers [Lue66] and of many adaptive schemes [Lan79]. Other estimators, such as maximum-likelihood estimators [KS67] or the ubiquitous Kalman filter [Kal60] [Sor83], are based on a statistical description of uncertainty, and assume that the measurement noise and state perturbations are realizations of random variables, with known statistical properties. A last group of methods, known under the generic names of *set-membership estimation* or *guaranteed estimation* or *bounded-error estimation*, see, *e.g.*, [Wal90], [Nor94], [Nor95], [MNPLW96] and the references therein, is based on the assumption that the uncertain variables (such as noise and perturbations) all belong to known compact sets, and attempts to build simple sets, such as ellipsoids, orthotopes or parallelotopes, guaranteed to contain the vectors to be estimated.

In a nonlinear context, the methodology is far less developed, and still the subject of active research even in the deterministic case [KK98]. When uncertainty is explicitly taken into account, this is usually by resorting to linearization. For parameter estimation, one may exploit the asymptotic properties of maximum-likelihood estimators, but the validity of the results obtained from a small data base is then questionable. For state estimation, an extended Kalman filter [Gel74], based on a linearization of the model around its trajectory, is usually employed. This linearization is inherently local and may fail to produce reliable estimates. It makes any statistical interpretation of the covariance matrices computed by the algorithm questionable, because the propagation of the statistical properties of the perturbations through the nonlinear system is largely unknown. As far as set-membership estimation is concerned, very few guaranteed methods are available, most of them developed for parameter estimation. They are based on branch-and-bound techniques (see, *e.g.,* [MV91] for a signomial programming approach and [JW93] for an interval computation approach). A method based on interval analysis to compute guaranteed state estimates was proposed in [KJW98] and [KJWM99].

The purpose of this paper is to present a new approach for the guaranteed estimation of the parameter and/or state vector of a nonlinear discrete-time model in a bounded-error context. The approach uses the ideas of *interval constraint propagation (ICP)* (see, *e.g.*, [Dav87], [Cle87], [Hyv92], [AFL00]), which combines interval analysis and *constraint satisfaction problems (CSP)* (see [MR91]). ICP has already been used for solving bounded-error estimation problems in [Jau00] and [Jau01]. Consider a nonlinear discrete-time system described by

$$\left\{ \begin{array}{rcl} \mathbf{x}_k & = & \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{p}, \mathbf{w}_{k-1}, \mathbf{u}_{k-1}) \\ \mathbf{y}_k & = & \mathbf{g}_k(\mathbf{x}_k, \mathbf{p}, \mathbf{w}_k, \mathbf{u}_k) \end{array} \right. \qquad k = 1, \ldots, \bar{k}, \tag{1}$$

where k is the time index, $\mathbf{x}_k$ is the state vector, $\mathbf{y}_k$ is the output vector, $\mathbf{u}_k$ is the input vector, $\mathbf{w}_k$ is the perturbation vector, $\mathbf{p}$ is a constant parameter vector and $\mathbf{f}_k$ and $\mathbf{g}_k$ are known functions.

Remark 1 *Equation (1) allows the usual distinction between a state perturbation* $\mathbf{w}_k^x$ *and a measurement noise* $\mathbf{w}_k^y$ *as a special case, since it suffices to define* $\mathbf{w}_k$ *as the concatenation of* $\mathbf{w}_k^x$ *and* $\mathbf{w}_k^y$. $\diamondsuit$

The set of all variables involved in this problem is

$$\mathcal{V} = \{\mathbf{p}, \mathbf{x}_0, \mathbf{w}_0, \mathbf{u}_0, \mathbf{x}_1, \mathbf{w}_1, \mathbf{u}_1, \mathbf{y}_1, \ldots, \mathbf{x}_{\bar{k}}, \mathbf{w}_{\bar{k}}, \mathbf{u}_{\bar{k}}, \mathbf{y}_{\bar{k}}\}. \tag{2}$$

We shall assume that there exist some unknown actual values**,** denoted by $\mathbf{x}_k^*, \mathbf{w}_k^*, \mathbf{u}_k^*, \mathbf{y}_k^*$ and $\mathbf{p}^*$, for $\mathbf{x}_k, \mathbf{w}_k, \mathbf{u}_k, \mathbf{y}_k$ and $\mathbf{p}$, such that (1) is satisfied. This assumption will allow us to interpret the estimation problem as that of finding reliable estimates for these actual values, but note that it is not required for the application of the method.

The set-membership approach to be followed in this paper characterizes the uncertainty about the actual value $\mathbf{v}^*$ of any given variable $\mathbf{v} \in \mathcal{V}$ by associating with $\mathbf{v}$ a *domain* $\mathbb{V}$ that contains $\mathbf{v}^*$. The set of all such domains is

$$\mathcal{D} = \{\mathbb{P}, \mathbb{X}_0, \mathbb{W}_0, \mathbb{U}_0, \mathbb{X}_1, \mathbb{W}_1, \mathbb{U}_1, \mathbb{Y}_1, \ldots, \mathbb{X}_{\bar{k}}, \mathbb{W}_{\bar{k}}, \mathbb{U}_{\bar{k}}, \mathbb{Y}_{\bar{k}}\}. \tag{3}$$

When the actual value of a variable $\mathbf{v}$ is known exactly, $\mathbb{V}$ is the singleton $\{\mathbf{v}^*\}$. When nothing is known about $\mathbf{v}^*$, $\mathbb{V} = \mathbb{R}^{\dim \mathbf{v}}$.

A measurement $\mathbf{v}^{\mathrm{m}}$ of a variable $\mathbf{v} \in \mathcal{V}$ provides an approximation of $\mathbf{v}^*$. Let $\mathcal{P}\left(\mathbb{R}^{\dim \mathbf{v}}\right)$ be the set of all subsets of $\mathbb{R}^{\dim \mathbf{v}}$. We shall call *interpretation function* associated with $\mathbf{v}$, any set-valued function $\phi_{\mathbf{v}} : \mathbb{R}^{\dim \mathbf{v}} \to \mathcal{P}\left(\mathbb{R}^{\dim \mathbf{v}}\right)$ that satisfies $\mathbf{v}^* \in \phi_{\mathbf{v}}(\mathbf{v}^{\mathrm{m}})$. In practice, the effect of this function is to inflate $\mathbf{v}^{\mathrm{m}}$ to take the measurement error into account. The set $\phi_{\mathbf{v}}(\mathbf{v}^{\mathrm{m}})$ is the *measurement uncertainty set*. As soon as $\mathbf{v}^{\mathrm{m}}$ is made available, the domain $\mathbb{V}$ for $\mathbf{v}$ can be replaced by $\mathbb{V} \cap \phi_{\mathbf{v}}(\mathbf{v}^{\mathrm{m}})$. In (1), only the variables $\mathbf{u}_k$ and $\mathbf{y}_k$ are assumed to be measured. Two situations will be distinguished:

- A *causal context*: at time k, the measurements are available up to time k only, *i.e.*, the available data are $\{\mathbf{u}_0^{\mathrm{m}}, \mathbf{u}_1^{\mathrm{m}}, \mathbf{y}_1^{\mathrm{m}}, \ldots, \mathbf{u}_k^{\mathrm{m}}, \mathbf{y}_k^{\mathrm{m}}\}$.
- A *noncausal context*: all measurements $\{\mathbf{u}_0^{\mathrm{m}}, \mathbf{u}_1^{\mathrm{m}}, \mathbf{y}_1^{\mathrm{m}}, \ldots, \mathbf{u}_k^{\mathrm{m}}, \mathbf{y}_k^{\mathrm{m}}, \ldots, \mathbf{u}_{\bar{k}}^{\mathrm{m}}, \mathbf{u}_{\bar{k}}^{\mathrm{m}}\}$ are available from the start.

This distinction is similar to that between estimating and smoothing Kalman filters. Even in a causal context, some prior information may be available on variables before any measurement

is collected. For instance, physical constraints may provide upper and lower bounds on some components of vector variables, which can then be taken into account in the definition of the corresponding domains. The measurements and the constraints associated with the $2\bar{k}$ equations given by (1) will be used to reduce the domains and thus the uncertainty on the variables, which can be formulated as a *generalized set estimation problem*.

The basic step of this generalized set estimation problem is to find the smallest domains $\widehat{\mathbb{P}}$, $\widehat{\mathbb{X}}_0$, $\widehat{\mathbb{W}}_0$, $\widehat{\mathbb{U}}_0$, $\widehat{\mathbb{X}}_1$, $\widehat{\mathbb{W}}_1$, $\widehat{\mathbb{U}}_1$, $\widehat{\mathbb{Y}}_1, \ldots, \widehat{\mathbb{X}}_{\bar{k}}$, $\widehat{\mathbb{W}}_{\bar{k}}$, $\widehat{\mathbb{U}}_{\bar{k}}$ and $\widehat{\mathbb{Y}}_{\bar{k}}$ such that the following implication is satisfied

$$\left\{\begin{array}{l} \text{(1) hold true and} \\ \mathbf{p} \in \mathbb{P} \\ \mathbf{x}_0 \in \mathbb{X}_0, \ldots, \mathbf{x}_{\bar{k}} \in \mathbb{X}_{\bar{k}} \\ \mathbf{w}_0 \in \mathbb{W}_0, \ldots, \mathbf{w}_{\bar{k}} \in \mathbb{W}_{\bar{k}} \\ \mathbf{u}_0 \in \mathbb{U}_0, \ldots, \mathbf{u}_{\bar{k}} \in \mathbb{U}_{\bar{k}} \\ \mathbf{y}_1 \in \mathbb{Y}_1, \ldots, \mathbf{y}_{\bar{k}} \in \mathbb{Y}_{\bar{k}} \end{array}\right. \Rightarrow \left\{\begin{array}{l} \mathbf{p} \in \widehat{\mathbb{P}} \\ \mathbf{x}_0 \in \widehat{\mathbb{X}}_0, \ldots, \mathbf{x}_{\bar{k}} \in \widehat{\mathbb{X}}_{\bar{k}} \\ \mathbf{w}_0 \in \widehat{\mathbb{W}}_0, \ldots, \mathbf{w}_{\bar{k}} \in \widehat{\mathbb{W}}_{\bar{k}} \\ \mathbf{u}_0 \in \widehat{\mathbb{U}}_0, \ldots, \mathbf{u}_{\bar{k}} \in \widehat{\mathbb{U}}_{\bar{k}} \\ \mathbf{y}_1 \in \widehat{\mathbb{Y}}_1, \ldots, \mathbf{y}_{\bar{k}} \in \widehat{\mathbb{Y}}_{\bar{k}} \end{array}\right. \tag{4}$$

Since $\widehat{\mathbb{V}} \subset \mathbb{V}$, $\widehat{\mathbb{V}}$ can now replace $\mathbb{V}$ as a more accurate domain for $\mathbf{v}$. The operation thus performed can be written concisely as the instruction

$$\mathcal{D} := \mathcal{E}(\mathcal{D}); \tag{5}$$

The *generalized set estimator* based on (5) can then be sketched as follows:

Initialization	set $\mathcal{D}$ as specified by available information;
1	reduce all domains by $\mathcal{D} := \mathcal{E}(\mathcal{D})$;
2	wait for a new measurement $\mathbf{v}^{\mathrm{m}}$;
3	in $\mathcal{D}$, replace $\mathbb{V}$ by $\mathbb{V} \cap \phi_{\mathbf{v}}(\mathbf{v}^{\mathrm{m}})$;
4	go to Step 1;

Note that in a noncausal context, all measurements are given at the initialization. Therefore, the estimation process stops after the first execution of Step 1.

Set parameter estimation, parameter tracking, state estimation and joint state and parameter estimation can all be seen as special cases of generalized set estimation. Moreover, the problem to be solved at Step 1 is itself a special case of a Set Constraint Satisfaction Problem (SCSP), to be presented in Section 2 in a more general context. In Section 3, constraint propagation techniques will be used to derive new set algorithms able to solve a large class of SCSPs, which includes the problem of Step 1. This general approach will be applied in Section 4 to causal and noncausal state estimation, and Section 5 will present an illustrative example.

2 Set Constraint Satisfaction Problems

This section presents some basic definitions and algorithms that are classical in the area of constraint propagation [Dav87], [Cle87], [Hyv92], but here these definitions and algorithms are extended to the case where intervals are replaced by more general subsets of real vector spaces.

Let $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ be a finite set of vector variables with dimensions $d_1 \in \mathbb{N}, \ldots, d_n \in \mathbb{N}$ and domains $\mathbb{V}_1 \subset \mathbb{R}^{d_1}, \ldots, \mathbb{V}_n \subset \mathbb{R}^{d_n}$. The *global space* is the set $\mathbb{R}^d = \mathbb{R}^{d_1} \times \cdots \times \mathbb{R}^{d_n}$, where $d = d_1 + \cdots + d_n$. The *global domain* is the set $\mathbb{V} = \mathbb{V}_1 \times \cdots \times \mathbb{V}_n$. Of course, $\mathbb{V} \subset \mathbb{R}^d$. The subscript i of a variable $\mathbf{v}_i$ and associated domain $\mathbb{V}_i$ will be called their *index*.

Let $\mathbf{v}_i$ and $\mathbf{v}_j$ be two elements of $\mathcal{V}$. A *binary constraint* $\mathbb{C}_{i,j}$ over $\mathbf{v}_i$ and $\mathbf{v}_j$ is a subset of $\mathbb{R}^{d_i} \times \mathbb{R}^{d_j}$. Often this constraint can be put in the form

$$\mathbb{C}_{i,j} = \left\{ (\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_j) \in \mathbb{R}^{d_i} \times \mathbb{R}^{d_j} \mid \tilde{\mathbf{v}}_j = \mathbf{f}_j (\tilde{\mathbf{v}}_i) \right\}. \tag{6}$$

We shall then refer to it as $\mathbb{C}_{i,j} : \mathbf{v}_j = \mathbf{f}_j (\mathbf{v}_i)$. In this paper, we shall only consider binary constraints. Note that n-ary constraints with $n > 2$ can always be decomposed into a set of binary constraints, so this is not limitative.

A *Set Constraint Satisfaction Problem* (SCSP) is a 3-uple $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$, where $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ is a finite set of vector variables, $\mathcal{D} = \{\mathbb{V}_1, \ldots, \mathbb{V}_n\}$ is the set of their domains and $\mathcal{C}$ is a finite set of binary constraints relating variables of $\mathcal{V}$.

Example 1 *Consider three real numbers x, y and z related by the constraint $z = x^2 + y^2$. Assume that z is known to belong to the interval $[-3, 1]$. This situation can be represented by an SCSP $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$, with $\mathcal{V} = \{\mathbf{v}_1, v_2\}, \mathcal{D} = \{\mathbb{V}_1, \mathbb{V}_2\}, \mathcal{C} = \{\mathbb{C}_{1,2}\}$, $\mathbf{v}_1 = (x \ y)^T, v_2 = z, \mathbb{V}_1 = \mathbb{R}^2, \mathbb{V}_2 = [-3, 1]$ and $\mathbb{C}_{1,2} : z = x^2 + y^2$. Note that if real variables were used instead of vector variables, it would not be possible to transform the ternary constraint ($z = x^2 + y^2$) into a binary constraint $v_2 = f(\mathbf{v}_1)$, where $f : \mathbb{R}^2 \longrightarrow \mathbb{R}; \ (x, y) \longrightarrow x^2 + y^2$.* $\diamondsuit$

A *point solution* of $\mathcal{H}$ is an n-uple $(\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_n) \in \mathbb{V} \subset \mathbb{R}^d$ such that for all constraints $\mathbb{C}_{i,j} \in \mathcal{C}$, the pair $(\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_j) \in \mathbb{C}_{i,j}$. The set of all point solutions of $\mathcal{H}$ is denoted by $\mathbb{S}(\mathcal{H})$. This set will be called the *global solution set*. In Example 1, $\mathbb{S} = \{(x, y, z) \,|z = x^2 + y^2 \text{ and } z \in [-3, 1]\}$.

The variable $\mathbf{v}_i$ is *consistent* in $\mathcal{H}$ (or $\mathcal{H}$-consistent) if

$$\forall \tilde{\mathbf{v}}_i \in \mathbb{V}_i, \exists (\tilde{\mathbf{v}}_1 \in \mathbb{V}_1, \ldots, \tilde{\mathbf{v}}_{i-1} \in \mathbb{V}_{i-1}, \tilde{\mathbf{v}}_{i+1} \in \mathbb{V}_{i+1}, \ldots, \tilde{\mathbf{v}}_n \in \mathbb{V}_n) \mid (\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_n) \in \mathbb{S}(\mathcal{H}). \tag{7}$$

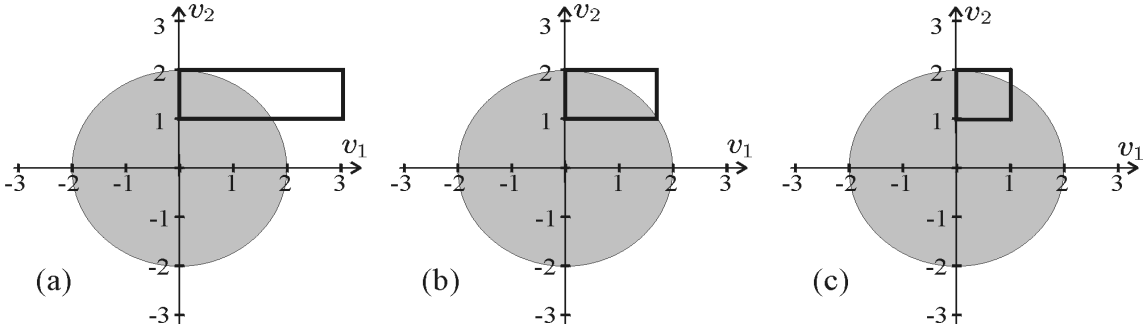

Figure 1: Illustration of the SCSPs $\mathcal{H}_{\mathrm{a}}$, $\mathcal{H}_{\mathrm{b}}$ and $\mathcal{H}_{\mathrm{c}}$

Example 2 *Consider three SCSPs $\mathcal{H}_a, \mathcal{H}_b$ and $\mathcal{H}_c$. Each of them involves two real variables v_1 and v_2 and the single constraint $v_1^2 + v_2^2 \leq 4$. They only differ by their domains, given by*

	$\mathbb{V}_1$	$\mathbb{V}_2$
$\mathcal{H}_a$	$[0, 3]$	$[1, 2]$
$\mathcal{H}_b$	$[0, \sqrt{3}]$	$[1, 2]$
$\mathcal{H}_c$	$[0, 1]$	$[1, 2]$

(8)

as illustrated by Figure 1. The variable v_2 is $\mathcal{H}_a, \mathcal{H}_b$ and $\mathcal{H}_c$-consistent because for all $v_2 \in \mathbb{V}_2 = [1, 2]$, there exists $v_1 \in \mathbb{V}_1$ such that $v_1^2 + v_2^2 \leq 4$. The variable v_1 is $\mathcal{H}_b$ and $\mathcal{H}_c$-consistent but not $\mathcal{H}_a$-consistent. For the three SCSPs, the global solution set is given by the intersection of the disk (for the constraint) and the box $\mathbb{V}_1 \times \mathbb{V}_2$. Note that $\mathbb{S}(\mathcal{H}_a) = \mathbb{S}(\mathcal{H}_b) \supset \mathbb{S}(\mathcal{H}_c)$. $\diamondsuit$

In Example 1, neither $\mathbf{v}_1$ nor v_2 is $\mathcal{H}$-consistent. If the domain $\mathbb{V}_1$ is replaced by the disk centered at 0 and with a radius a in $[0, 1]$, denoted by $\mathrm{Disk}(0, a)$ then $\mathbf{v}_1$ becomes $\mathcal{H}$-consistent. If $\mathbb{V}_2$ is replaced by the interval $[0, a^2]$, then v_2 becomes $\mathcal{H}$-consistent. Note that for a given SCSP $\mathcal{H}$, if $\mathbf{v}_i$ is $\mathcal{H}$-consistent and if its domain $\mathbb{V}_i$ is replaced by any subset of $\mathbb{V}_i$, then $\mathbf{v}_i$ is still consistent in the new SCSP.

If $\mathcal{I} = \{i_1, \ldots, i_p\}$ is a subset of the set of indices $\{1, \ldots, n\}$, then $\mathcal{H}' = (\mathcal{V}', \mathcal{D}', \mathcal{C}')$, where $\mathcal{V}' \triangleq \{\mathbf{v}_{i_1}, \ldots, \mathbf{v}_{i_p}\}$, $\mathcal{D}' \triangleq \{\mathbb{V}_{i_1}, \ldots, \mathbb{V}_{i_p}\}$ and $\mathcal{C}' \triangleq \{\mathbb{C}_{i,j} \in \mathcal{C} \text{ such that } i \in \mathcal{I} \text{ and } j \in \mathcal{I}\}$, is called a *subSCSP* of $\mathcal{H}$. It is trivial to show that if $\mathbf{v}_i \in \mathcal{V}'$ is $\mathcal{H}$-consistent, it is also $\mathcal{H}'$-consistent.

The ith *projected domain* $\mathbb{S}_i = \pi_i(\mathcal{H})$ onto the variable $\mathbf{v}_i$ is the largest domain $\mathbb{S}_i \subset \mathbb{V}_i$ such that if $\mathbb{V}_i$ is replaced by $\mathbb{S}_i$ in $\mathcal{H}$, $\mathbf{v}_i$ becomes $\mathcal{H}$-consistent. It can also be defined by the

orthogonal projection of the global solution set $\mathbb{S}(\mathcal{H})$ onto $\mathbb{R}^{d_i}$, *i.e.*,

$$\begin{array}{ll} & \tilde{\mathbf{v}}_i \in \pi_i(\mathcal{H}) \\ \Leftrightarrow & \exists (\tilde{\mathbf{v}}_1 \in \mathbb{V}_1, \ldots, \tilde{\mathbf{v}}_{i-1} \in \mathbb{V}_{i-1}, \tilde{\mathbf{v}}_{i+1} \in \mathbb{V}_{i+1}, \ldots, \tilde{\mathbf{v}}_n \in \mathbb{V}_n) \mid (\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_n) \in \mathbb{S}(\mathcal{H}). \end{array} \tag{9}$$

Note that if $\mathbf{v}_i$ is $\mathcal{H}$-consistent, $\mathbb{V}_i = \pi_i(\mathcal{H})$. In Example 2, $\pi_1(\mathcal{H}_\mathrm{a}) = \pi_1(\mathcal{H}_\mathrm{b}) = [0, \sqrt{3}]$, $\pi_1(\mathcal{H}_\mathrm{c}) = [0,1]$ and $\pi_2(\mathcal{H}_\mathrm{a}) = \pi_2(\mathcal{H}_\mathrm{b}) = \pi_2(\mathcal{H}_\mathrm{c}) = [1,2]$. In Example 1, $\pi_1(\mathcal{H}) = \mathrm{Disk}(0,1)$ and $\pi_2(\mathcal{H}) = [0,1]$.

$\mathcal{H}$ is an *elementary SCSP* if $\mathcal{V}$ is a singleton $\{\mathbf{v}_1\}$. Therefore, $\mathcal{C} = \emptyset$ and $\mathbb{V}_1$ is necessarily $\mathcal{H}$-consistent.

Two SCSPs $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$ and $\mathcal{H}' = (\mathcal{V}', \mathcal{D}', \mathcal{C}')$ are *equivalent,* denoted by $\mathcal{H} \equiv \mathcal{H}'$, if $\mathcal{V} = \mathcal{V}', \mathcal{C} = \mathcal{C}'$ and $\mathbb{S}(\mathcal{H}) = \mathbb{S}(\mathcal{H}')$. We shall say that $\mathcal{H}'$ is a *contraction* of $\mathcal{H}$ if $\mathcal{H}' \equiv \mathcal{H}$ and $\mathbb{V}'_i \subset \mathbb{V}_i$ for any index i. To *contract* an SCSP is to replace it by one of its contractions. $\mathcal{H}$ is *minimal* if it admits no contraction of itself but itself. In Example 2, we have $\mathcal{H}_\mathrm{a} \equiv \mathcal{H}_\mathrm{b}$; $\mathcal{H}_\mathrm{b}$ is a contraction of $\mathcal{H}_\mathrm{a}$ and $\mathcal{H}_\mathrm{b}$ and $\mathcal{H}_\mathrm{c}$ are minimal.

$\mathcal{H}'$ is the *optimal contraction* of $\mathcal{H}$ if $\mathcal{H}' \equiv \mathcal{H}$ and $\mathcal{H}'$ is minimal. We shall write $\mathcal{H}' = \mathcal{E}(\mathcal{H})$ or sometimes $\mathcal{D}' = \mathcal{E}(\mathcal{D})$ as in (5), since $\mathcal{H}$ and $\mathcal{H}'$ can only differ by their domains (in Example 2, $\mathcal{H}_\mathrm{b} = \mathcal{E}(\mathcal{H}_\mathrm{a})$). The n domains $\mathbb{V}'_i$ of the optimal contraction satisfy $\mathbb{V}'_i = \pi_i(\mathcal{H}), i \in \{1, \ldots, n\}$.

Consider two variables $\mathbf{v}_i$ and $\mathbf{v}_j$ related by a constraint $\mathbb{C}_{i,j}$. The *local contraction operator* of the domain $\mathbb{V}_i$ with respect to the variable $\mathbf{v}_j$ is defined as

$$\rho_j(\mathbb{V}_i) = \{\tilde{\mathbf{v}}_i \in \mathbb{V}_i | \exists \tilde{\mathbf{v}}_j \in \mathbb{V}_j, (\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_j) \in \mathbb{C}_{i,j}\}. \tag{10}$$

Note that $\rho_j(\mathbb{V}_i) \subset \mathbb{V}_i$. The new SCSP obtained by replacing $\mathbb{V}_i$ by $\mathbb{V}'_i = \rho_j(\mathbb{V}_i)$ is thus a contraction of the former SCSP. Figure 2 illustrates this definition. If the constraint $\mathbb{C}_{i,j}$ is given by $\mathbb{C}_{i,j} : \mathbf{v}_j = \mathbf{f}_j(\mathbf{v}_i)$, then

$$\rho_j(\mathbb{V}_i) = \mathbb{V}_i \cap \mathbf{f}_j^{-1}(\mathbb{V}_j), \tag{11}$$

$$\rho_i(\mathbb{V}_j) = \mathbb{V}_j \cap \mathbf{f}_j(\mathbb{V}_i). \tag{12}$$

In Example 1, if $f(x,y) = x^2 + y^2$, then

$$\rho_1(\mathbb{V}_2) = [-3,1] \cap f\left(\mathbb{R}^2\right) = [-3,1] \cap [0,\infty[= [0,1], \tag{13}$$

$$\rho_2(\mathbb{V}_1) = \mathbb{R}^2 \cap f^{-1}([-3,1]) = \mathrm{Disk}(0,1). \tag{14}$$

The Waltz algorithm [Wal75], [Dav87] is one of the basic algorithms that can be used to perform contractions of SCSPs. Its principle is to choose any constraint in $\mathcal{C}$ and to contract the domains

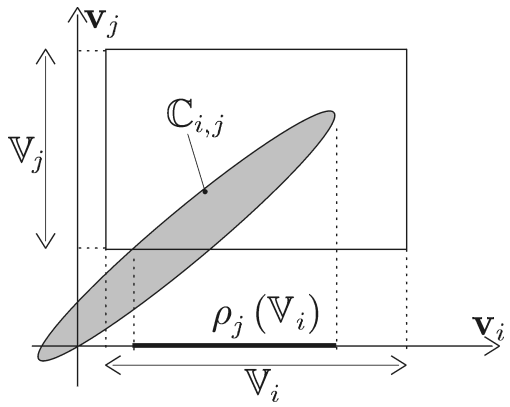

Figure 2: Local contraction operator

of the two associated variables $\mathbf{v}_i$ and $\mathbf{v}_j$ using the local contraction operators ρ_i and ρ_j. This is continued until no constraint in $\mathcal{C}$ is able to contract any domain. Unfortunately, the resulting SCSP may be non-minimal, because the algorithm may come to a deadlock, as illustrated by the following example.

Example 3 *Consider the SCSP* $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$*, where*

$$\begin{aligned} \mathcal{V} &= \{v_1, v_2, v_3\}, \\ \mathcal{D} &= \{\mathbb{V}_1 = [-1, 1]; \mathbb{V}_2 = [-1, 1]; \mathbb{V}_3 = [-1, 1]\}, \\ \mathcal{C} &= \{\mathbb{C}_{1,2} : v_2 = -v_1; \mathbb{C}_{2,3} : v_3 = -v_2; \mathbb{C}_{3,1} : v_1 = -v_3\}. \end{aligned} \tag{15}$$

Although the only solution is $v_1 = v_2 = v_3 = 0$*, since for all* $i, j \in \{1, 2, 3\}$*,* $i \neq j$*,* $\rho_i(\mathbb{V}_j) = \mathbb{V}_j$*, the Waltz algorithm is unable to contract any of domains* $\mathbb{V}_i$*.* $\diamondsuit$

The *graph* of any given SCSP $\mathcal{H}$ can be constructed as follows. To each variable $\mathbf{v}_i$, is associated a node and to each binary constraint $\mathbb{C}_{i,j}$ is associated an arc between the nodes $\mathbf{v}_i$ and $\mathbf{v}_j$. The graph associated with Example 3 is depicted on Figure 3. The failure of the Waltz algorithm to contract this SCSP is due to the fact that its graph contains a cycle.

When the graph is a tree, *i.e.,* a connected graph without cycles, the Waltz algorithm converges to the optimal contraction of $\mathcal{H}$. This result is a direct consequence of Theorem 6 in Section 3, which corresponds to a more efficient contraction algorithm.

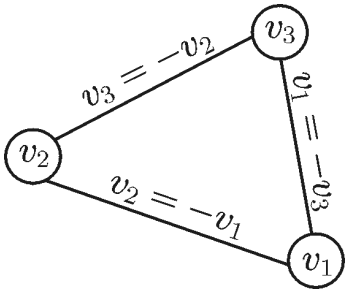

Figure 3: Graph of a SCSP with a cycle

3 Contraction algorithms

When the graph of the SCSP is a tree, the Waltz algorithm produces an optimal contraction, but not very efficiently because the constraints are taken into account in arbitrary order. In this section, we propose a new algorithm to contract an SCSP whose graph is a tree, which also produces an optimal contraction, but in a much more efficient way. It extends to vector variables the *propagation-retropropagation algorithm* proposed in [BGGP99] for real variables. It is based on the next two theorems.

3.1 Propagation and retropropagation theorems

Theorem 1 *(propagation theorem): Let* $\mathcal{H}_a = (\mathcal{V}_a, \mathcal{D}_a, \mathcal{C}_a)$ *and* $\mathcal{H}_b = (\mathcal{V}_b, \mathcal{D}_b, \mathcal{C}_b)$ *be two SCSPs with all their variables distinct. Let* $\mathbf{v}_{a1} \in \mathcal{V}_a$ *and* $\mathbf{v}_{b1} \in \mathcal{V}_b$ *be two variables related by the constraint* $\mathbb{C}_{a1,b1}$*. The new SCSP* $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$*, where*

$$\mathcal{V} = \mathcal{V}_a \cup \mathcal{V}_b;\ \mathcal{D} = \mathcal{D}_a \cup \mathcal{D}_b;\ \mathcal{C} = \mathcal{C}_a \cup \mathcal{C}_b \cup \{\mathbb{C}_{a1,b1}\},$$

is such that if $\mathbf{v}_{a1}$ *is* $\mathcal{H}_a$*-consistent and* $\mathbf{v}_{b1}$ *is* $\mathcal{H}_b$*-consistent, then* $\pi_{a1}(\mathcal{V}, \mathcal{D}, \mathcal{C}) = \rho_{b1}(\mathbb{V}_{a1})$ *and* $\pi_{b1}(\mathcal{V}, \mathcal{D}, \mathcal{C}) = \rho_{a1}(\mathbb{V}_{b1})$*.* $\diamondsuit$

Proof: Because of the symmetry of the problem, it suffices to prove that $\pi_{a1}(\mathcal{H}) = \rho_{b1}(\mathbb{V}_{a1})$. Let $\mathbf{v}_{a2}, \ldots, \mathbf{v}_{ap}$ be the $p-1$ variables of $\mathcal{V}_a$ distinct from $\mathbf{v}_{a1}$ and $\mathbf{v}_{b2}, \ldots, \mathbf{v}_{bq}$ be the $q-1$ variables of $\mathcal{V}_b$ distinct from $\mathbf{v}_{b1}$. Since $\mathbf{v}_{a1}$ is $\mathcal{H}_a$-consistent and $\mathbf{v}_{b1}$ is $\mathcal{H}_b$-consistent, $\pi_{a1}(\mathcal{H}_a) = \mathbb{V}_{a1}$ and $\pi_{b1}(\mathcal{H}_b) = \mathbb{V}_{b1}$. Now, $\tilde{\mathbf{v}}_{a1} \in \pi_{a1}(\mathcal{H}) \Leftrightarrow \tilde{\mathbf{v}}_{a1} \in \mathbb{V}_{a1}$ and $\exists \tilde{\mathbf{v}}_{a2} \in \mathbb{V}_{a2}, \ldots, \exists \tilde{\mathbf{v}}_{ap} \in \mathbb{V}_{ap}, \exists \tilde{\mathbf{v}}_{b1} \in \mathbb{V}_{b1}$,

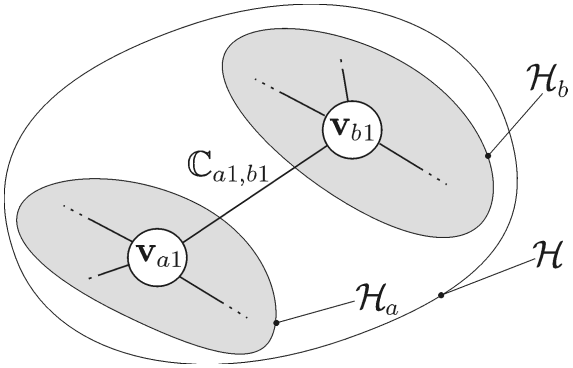

Figure 4: The three SCSPs involved in the propagation theorem

$\ldots$, $\exists \tilde{\mathbf{v}}_{bn} \in \mathbb{V}_{bn}$ such that all constraints of $\mathcal{C}$ are satisfied, *i.e.*,

$$\begin{array}{l} \tilde{\mathbf{v}}_{a1} \in \pi_{a1}(\mathcal{H}) \\ \Leftrightarrow \left\{ \begin{array}{l} \text{(i) } \exists \tilde{\mathbf{v}}_{a2} \in \mathbb{V}_{a2}, \ldots, \exists \tilde{\mathbf{v}}_{ap} \in \mathbb{V}_{ap} | \, (\tilde{\mathbf{v}}_{a1}, \ldots, \tilde{\mathbf{v}}_{ap}) \in \mathbb{S}(\mathcal{H}_a) \\ \text{(ii) } \exists \tilde{\mathbf{v}}_{b1} \in \mathbb{V}_{b1}, \ldots, \exists \tilde{\mathbf{v}}_{bn} \in \mathbb{V}_{bn} \mid \left\{ \begin{array}{c} (\tilde{\mathbf{v}}_{a1}, \tilde{\mathbf{v}}_{b1}) \in \mathbb{C}_{a1,b1} \\ (\tilde{\mathbf{v}}_{b1}, \ldots, \tilde{\mathbf{v}}_{bq}) \in \mathbb{S}(\mathcal{H}_b) \end{array} \right. \end{array} \right. \end{array} \tag{16}$$

Since $\mathbb{V}_{a1} = \pi_{a1}(\mathcal{H}_a)$, (i) is equivalent to $\tilde{\mathbf{v}}_{a1} \in \mathbb{V}_{a1}$. Since $\mathbb{V}_{b1} = \pi_{b1}(\mathcal{H}_b)$, (ii) is equivalent to $\exists \tilde{\mathbf{v}}_{b1} \in \mathbb{V}_{b1} | \, (\tilde{\mathbf{v}}_{a1}, \tilde{\mathbf{v}}_{b1}) \in \mathbb{C}_{a1,b1}$. The equivalence (16) becomes

$$\tilde{\mathbf{v}}_{a1} \in \pi_{a1}(\mathcal{H}) \Leftrightarrow \left\{ \begin{array}{l} \tilde{\mathbf{v}}_{a1} \in \mathbb{V}_{a1} \\ \exists \tilde{\mathbf{v}}_{b1} \in \mathbb{V}_{b1} | \, (\tilde{\mathbf{v}}_{a1}, \tilde{\mathbf{v}}_{b1}) \in \mathbb{C}_{a1,b1} \end{array} \right\} \Leftrightarrow \tilde{\mathbf{v}}_{a1} \in \rho_{b1}(\mathbb{V}_{a1}). \tag{17}$$

$\diamondsuit$

Figure 4 illustrates how the two former SCSPs $\mathcal{H}_a$ and $\mathcal{H}_b$ related by the constraint $\mathbb{C}_{a1,b1}$ form the new SCSP $\mathcal{H}$.

Corollary 2 *Let $\mathcal{H}_b = (\mathcal{V}_b, \mathcal{D}_b, \mathcal{C}_b)$ be an SCSP and $\mathbf{v}_a$ be a new variable with domain $\mathbb{V}_a$ related to a variable $\mathbf{v}_{b1} \in \mathcal{V}_b$ by the constraint $\mathbb{C}_{a,b1}$. In the SCSP $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$, where*

$$\mathcal{V} = \mathcal{V}_b \cup \{\mathbf{v}_a\}; \; \mathcal{D} = \mathcal{D}_b \cup \{\mathbb{V}_a\}; \; \mathcal{C} = \mathcal{C}_b \cup \{\mathbb{C}_{a,b1}\}, \tag{18}$$

if $\mathbf{v}_{b1}$ is $\mathcal{H}_b$-consistent then $\pi_a(\mathcal{H}) = \rho_{b1}(\mathbb{V}_a)$.

Proof: Consider the elementary SCSP $\mathcal{H}_a = (\mathcal{V}_a, \mathcal{D}_a, \mathcal{C}_a)$ where $\mathcal{V}_a = \{\mathbf{v}_a\}, \mathcal{D}_a = \{\mathbb{V}_a\}$ and $\mathcal{C}_a = \emptyset$. Theorem 1 implies that $\pi_a(\mathcal{H}) = \rho_{b1}(\mathbb{V}_a)$. $\diamondsuit$

Theorem 3 *(retropropagation theorem): Let* $\mathcal{H}_a = (\mathcal{V}_a, \mathcal{D}_a, \mathcal{C}_a)$ *and* $\mathcal{H}_b = (\mathcal{V}_b, \mathcal{D}_b, \mathcal{C}_b)$ *be two SCSPs with all their variables distinct. Let* $\mathbf{v}_{a1} \in \mathcal{V}_a$ *and* $\mathbf{v}_{b1} \in \mathcal{V}_b$ *be two variables related by the constraint* $\mathbb{C}_{a1,b1}$*. Consider the SCSP* $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$:

$$\mathcal{V} = \mathcal{V}_a \cup \mathcal{V}_b;\ \mathcal{D} = \mathcal{D}_a \cup \mathcal{D}_b;\ \mathcal{C} = \mathcal{C}_a \cup \mathcal{C}_b \cup \{\mathbb{C}_{a1,b1}\}. \tag{19}$$

If $\mathbf{v}_{a1}$ *is* $\mathcal{H}$*-consistent and* $\mathbf{v}_{b1}$ *is* $\mathcal{H}_b$*-consistent, then*

$$\pi_{b1}(\mathcal{H}) = \rho_{a1}(\mathbb{V}_{b1}). \tag{20}$$

♢

Proof: Since $\mathcal{H}_a$ is a subSCSP of $\mathcal{H}$ and since $\mathbf{v}_{a1}$ is $\mathcal{H}$-consistent, $\mathbf{v}_{a1}$ is also $\mathcal{H}_a$-consistent. From Theorem 1, $\pi_{b1}(\mathcal{V}, \mathcal{D}, \mathcal{C}) = \rho_{a1}(\mathbb{V}_{b1})$. ♢

3.2 Algorithms FALL and CLIMB

This section describes two efficient contraction algorithms that can be used to contract an SCSP $\mathcal{H}$ optimally, when its graph is a tree. FALL, based on the propagation theorem, scans the tree from its leaves down to its root and CLIMB, based on the retropropagation theorem, scans it from its root up to its leaves. It will be shown that a single execution of FALL followed by a single execution of CLIMB leads to the optimal contraction of $\mathcal{H}$. The same idea developed for discrete domains can be found in [MR91].

Remark 2 *Any SCSP containing cycles can be transformed into an equivalent SCSP whose graph is a tree. It suffices for that to group all variables responsible for the existence of a cycle into a single vector variable. Consider for instance the SCSP of Example 3 whose graph is given by Figure 3. If we set* $w_1 = v_1$ $\mathbf{w}_2 = (v_2, v_3)^T$, $\mathbb{W}_1 = [-1, 1]$, $\mathbb{W}_2 = \{(v_2, -v_2)^T \mid v_2 = [-1, 1]\}$ *and* $\mathbb{C}_{1,2} : (v_1 + v_3)^2 + (v_1 + v_2)^2 = 0$*, we get a SCSP with two nodes and one arc. Its graph is thus a tree.* ♢

To present FALL and CLIMB, some definitions concerning trees will be needed.

3.2.1 Trees

The trees to be considered are not directed. A *rooted tree* is obtained after selecting any node of a given tree $\mathcal{T}$ as its root. By analogy with the forest variety, we shall say that the root is

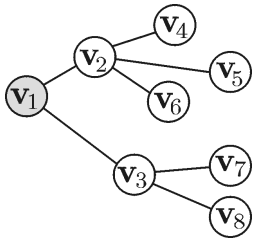

Figure 5: Example of a tree with eight nodes; the root chosen is indicated in grey

the lowest node of the tree. A node $\mathbf{v}_b$ is *above* a node $\mathbf{v}_a$ (or equivalently $\mathbf{v}_a$ is *below* $\mathbf{v}_b$) if the subtree $\mathcal{T}_a$ of $\mathcal{T}$ with root $\mathbf{v}_a$ contains $\mathbf{v}_b$. Consider a tree $\mathcal{T}$, with root $\mathbf{v}_1$. By removing $\mathbf{v}_1$, one obtains q subtrees $\mathcal{T}_{i_1}, \ldots, \mathcal{T}_{i_q}$ of $\mathcal{T}$. These q subtrees are the *branchs* of $\mathcal{T}$. A tree with no branch is a *leaf*. If $\mathcal{T}$ is a leaf, we shall write $\mathcal{T} = \mathbf{v}_1$. Otherwise, we shall write $\mathcal{T} = \mathbf{v}_1 \rightarrow \left(\mathcal{T}_{i_1} | \ldots | \mathcal{T}_{i_q}\right)$. Each branch of $\mathcal{T}$ will be rooted in a natural way by choosing its node that is connected to $\mathbf{v}_1$ as its root.

Example 4 *When* $\mathbf{v}_1$ *is chosen as its root, the tree* $\mathcal{T}$ *of Figure 5 will be denoted by*

$$\mathcal{T} = \mathbf{v}_1 \rightarrow (\mathbf{v}_2 \rightarrow (\mathbf{v}_4 | \mathbf{v}_5 | \mathbf{v}_6) \, | \mathbf{v}_3 \rightarrow (\mathbf{v}_7 | \mathbf{v}_8)) \, . \tag{21}$$

Its two branchs are $\mathcal{T}_2 = \mathbf{v}_2 \rightarrow (\mathbf{v}_4 | \mathbf{v}_5 | \mathbf{v}_6)$ *and* $\mathcal{T}_3 = \mathbf{v}_3 \rightarrow (\mathbf{v}_7 | \mathbf{v}_8)$ *.* $\mathcal{T}$ *has seven subtrees, five of which are leaves.* $\diamondsuit$

A *tree SCSP* (or *TSCSP*) $\mathcal{H}$ is an SCSP, the graph of which is a tree. A TSCSP can be rooted by selecting one of its node $\mathbf{v}_i$ as its root. Only rooted TSCSPs will be considered from now on, and we shall write $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C}, \mathbf{v}_i)$ to denote a TSCSP with root $\mathbf{v}_i$. All notions (root, leaves, branchs...) existing for trees extend to TSCSPs in a natural way.

Let $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C}, \mathbf{v}_i)$ be a TSCSP and $\mathbf{v}_j$ be an element of $\mathcal{V}$. Let $\mathcal{H}_j = \text{SubTree}(\mathcal{H}, \mathbf{v}_j)$ be the subTSCSP of $\mathcal{H}$ with root $\mathbf{v}_j$ associated with all nodes that are above $\mathbf{v}_j$. A variable $\mathbf{v}_j$ is *up-consistent* in $\mathcal{H}$ if it is consistent in $\mathcal{H}_j = \text{Subtree}(\mathcal{H}, \mathbf{v}_j)$. $\mathcal{H}$ is *up-consistent* if all of its nodes are up-consistent in $\mathcal{H}$.

3.2.2 Algorithm FALL

An up-consistent contraction of any TSCSP $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C}, \mathbf{v}_i)$ can be computed by the following recursive algorithm.

FALL(inout: $\mathcal{H}$)

1 if $\mathcal{H}$ is a leaf return;
2 $i := \text{index}(\text{root}(\mathcal{H}))$; $q :=$ number of branchs of $\mathcal{H}$;
3 for all branchs $\mathcal{H}_{j_k}$ of $\mathcal{H}$, $k \in \{1, \ldots, q\}$
4 FALL($\mathcal{H}_{j_k}$);
5 $j_k := \text{index}(\text{root}(\mathcal{H}_{j_k}))$;
6 $\mathbb{V}_i := \rho_{j_k}(\mathbb{V}_i)$;
7 endfor.

Theorem 4 *:* FALL *generates an up-consistent contraction of any TSCSP.*

Proof: The proof is in two parts. We shall first prove, by induction on k, that if FALL($\mathcal{H}_{j_k}$) makes $\mathcal{H}_{j_k}$ up-consistent at Step 4, then $\mathcal{H}$ is up-consistent after completion of FALL. Let $\mathcal{H}_i(k)$ be the subtree of $\mathcal{H}$, with root $\mathbf{v}_i$ and with k branchs $\mathcal{H}_{j_1}, \mathcal{H}_{j_2}, \ldots, \mathcal{H}_{j_k}$, in common with $\mathcal{H}$, $k \in \{1, \ldots, q\}$. For $k = 1$, from Corollary 2, $\mathcal{H}_i(1)$ is up-consistent after Step 6 (take $a = i$ and $b_1 = j_1$). Assume now that $\mathcal{H}_i(k)$ is up-consistent. From Theorem 1 (take $\mathcal{H}_a$ for the SCSP associated with $\mathcal{H}_i(k)$ and $b_1 = j_{k+1}$), $\mathcal{H}_i(k+1)$ is up-consistent after the $(k+1)$th execution of Step 6. Therefore $\mathcal{H} = \mathcal{H}_i(q)$ is up-consistent at Step 7. We shall now complete the proof by induction. If $\mathcal{H}$ is a leaf, it is already up-consistent and the theorem holds true. Assume that all branchs $\mathcal{H}_{j_1}, \ldots, \mathcal{H}_{j_q}$ of $\mathcal{H}$ have been made up-consistent by FALL. From the first part of the proof, FALL($\mathcal{H}$) then makes $\mathcal{H}$ up-consistent. ♢

Example 5 *Consider the TSCSP $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C}, \mathbf{v}_1)$ associated with the tree (21).* FALL *executes the following sequence of set operations*

$$\begin{array}{ll} \textit{(i)} & \mathbb{V}_2 := \rho_4(\mathbb{V}_2) \cap \rho_5(\mathbb{V}_2) \cap \rho_6(\mathbb{V}_2), \\ \textit{(ii)} & \mathbb{V}_3 := \rho_7(\mathbb{V}_3) \cap \rho_8(\mathbb{V}_3), \\ \textit{(iii)} & \mathbb{V}_1 := \rho_2(\mathbb{V}_1) \cap \rho_3(\mathbb{V}_1), \end{array} \tag{22}$$

which makes $\mathcal{H}$ up-consistent. The variable $\mathbf{v}_1$ is then also $\mathcal{H}$-consistent, i.e., $\mathbb{V}_1 = \pi_1(\mathcal{H})$. *Figure 6 illustrates this propagation assuming that the v_i's are real variables, the $\mathbb{V}_i$'s are all equal to $[-2, 2]$, and the domains defined by the constraints are represented in grey. Statement (i) corresponds to subfigures (b), (c) and (d) for $\rho_4(\mathbb{V}_2)$, $\rho_5(\mathbb{V}_2)$ and $\rho_6(\mathbb{V}_2)$ (which are represented by the vertical thick segments), respectively. The thick vertical segment of subfigure (a) is obtained by intersecting the thick vertical segments of (b), (c) and (d). Statement (ii) corresponds to (f) and (g) for $\rho_7(\mathbb{V}_3)$ and $\rho_8(\mathbb{V}_3)$, respectively. Statement (iii) corresponds to (a) and (e). The intersection of the thick horizontal segments of (a) and (e) represents the*

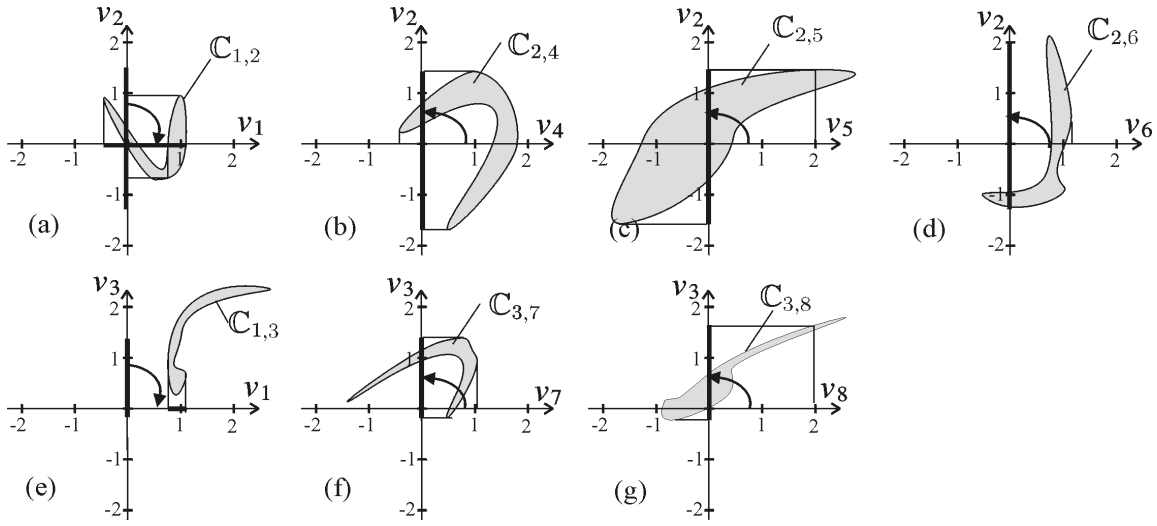

Figure 6: Illustration of propagation; the arrows indicate the direction of propagation

domain $\mathbb{V}_1$ *computed at (iii). Check that any other ordering of these operations would lead to more pessimistic domains.* ♢

3.2.3 Algorithm CLIMB

Let $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C}, \mathbf{v}_1)$ be an up-consistent TSCSP. This consistency may result from an execution of FALL($\mathcal{H}$). We shall now give an algorithm to compute $\mathcal{E}(\mathcal{H})$, the minimal contraction of $\mathcal{H}$. The principle of CLIMB is to propagate the $\mathcal{H}$-consistency of $\mathbf{v}_1$, from the root up to the leaves. One step of CLIMB is now described. Consider an $\mathcal{H}$-consistent variable $\mathbf{v}_i$ and one node $\mathbf{v}_j$ immediately above $\mathbf{v}_i$. Cut the arc between the nodes $\mathbf{v}_i$ and $\mathbf{v}_j$ (see Figure 7). Two TSCSPs are thus generated. One of them is $\mathcal{H}_j = (\mathcal{V}_j, \mathcal{D}_j, \mathcal{C}_j, \mathbf{v}_j) = \text{SubTree}(\mathcal{H}, \mathbf{v}_j)$. Since $\mathcal{H}$ is up-consistent, $\mathbf{v}_j$ is $\mathcal{H}_j$-consistent. From Theorem 3 (where $a1 = i, b1 = j$), $\pi_j(\mathcal{H}) = \rho_i(\mathbb{V}_j)$. This reasoning can be applied to all nodes of $\mathcal{H}$, from its root to its leaves. The following recursive algorithm is thus obtained.

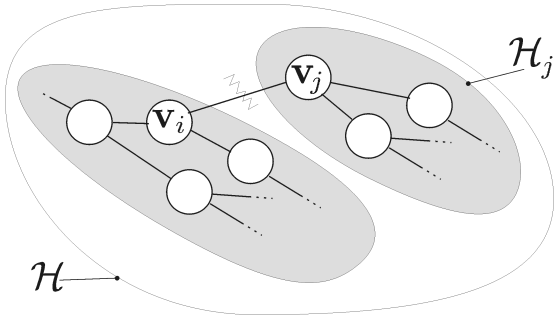

Figure 7: Illustration of the retropropagation step

```
        CLIMB(inout: H)
1   i := index(root(H));
2   for all branchs H_j of H
3       j := index(root(H_j));
4       V_j := ρ_i(V_j);
5       CLIMB(H_j);
6   endfor.
```

Theorem 5 CLIMB *computes the optimal contraction of any up-consistent TSCSP.* ♢

Proof: The proof is by induction. Since $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C}, \mathbf{v}_1)$ is up-consistent, $\mathbf{v}_1$ is $\mathcal{H}$-consistent. Assume now that the root $\mathbf{v}_i$ of the current subtree $\mathcal{H}_i$, is $\mathcal{H}$-consistent. Consider a branch $\mathbf{v}_j$ of $\mathbf{v}_i$. After Step 4, from Theorem 3 (where $a1 = i$ and $b1 = j$), $\mathbf{v}_j$ is $\mathcal{H}$-consistent. After completion of the algorithm, all variables are thus $\mathcal{H}$-consistent. ♢

Example 6 *Consider again the TSCSP $\mathcal{H}$ associated with the tree given by (21). Assume that $\mathcal{H}$ has been made up-consistent by* FALL *as in Example 5.* CLIMB *generates the following sequence of set operations: (a) $\mathbb{V}_2 := \rho_1(\mathbb{V}_2)$; (b) $\mathbb{V}_4 := \rho_2(\mathbb{V}_4)$; (c) $\mathbb{V}_5 := \rho_2(\mathbb{V}_5)$; (d) $\mathbb{V}_6 := \rho_2(\mathbb{V}_6)$; (e) $\mathbb{V}_3 := \rho_1(\mathbb{V}_3)$; (f) $\mathbb{V}_7 := \rho_3(\mathbb{V}_7)$; (g) $\mathbb{V}_8 := \rho_3(\mathbb{V}_8)$. From Theorem 5, $\mathcal{H}$ is now minimal. Figure 8 illustrates this retropropagation, which follows the propagation illustrated by Figure 6. The thicker segments correspond to the projected domains $\mathbb{S}_i = \pi_i(\mathcal{H})$. Note that the domains cannot be contracted anymore. This is due to the fact that the graph of $\mathcal{H}$ is a tree.* ♢

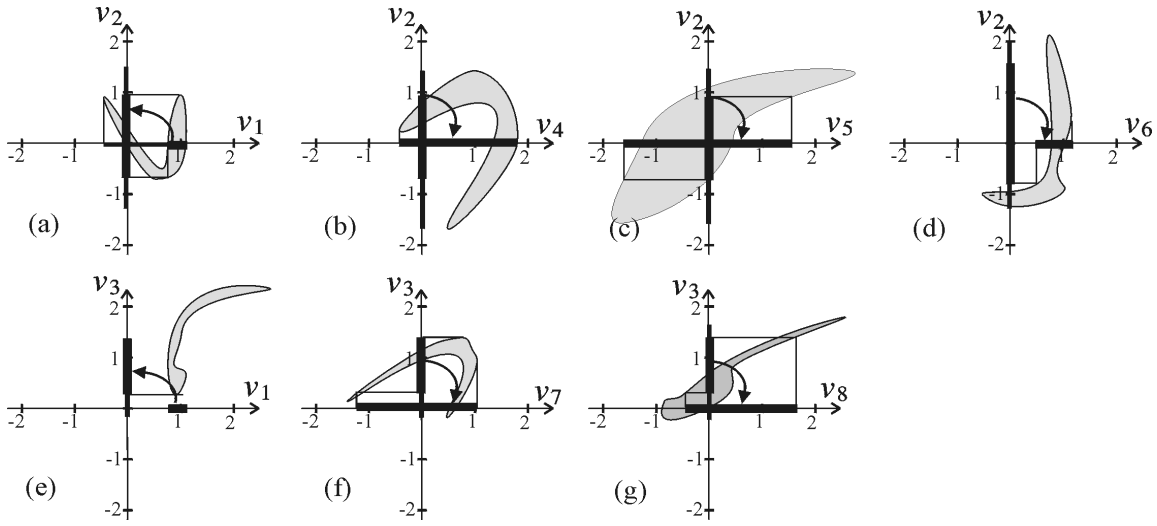

Figure 8: Illustration of retropropagation

3.2.4 Algorithm FALL-CLIMB

FALL and CLIMB can be combined to compute the minimal contraction of a given TSCSP $\mathcal{H}$. The resulting algorithm is:

FALL-CLIMB(inout: $\mathcal{H}$)
1 FALL($\mathcal{H}$);
2 CLIMB($\mathcal{H}$).

Theorem 6 FALL-CLIMB *returns the optimal contraction of any TSCSP.* ♢

Proof: Denote by $\mathcal{H}(0)$ the initial TSCSP, by $\mathcal{H}(1)$ the TSCSP after Step 1 and by $\mathcal{H}(2)$ the TSCSP after Step 2. From Theorem 4, $\mathcal{H}(1)$ is up-consistent and equivalent to $\mathcal{H}(0)$. From Theorem 5, $\mathcal{H}(2)$ is minimal and equivalent to $\mathcal{H}(0)$. ♢

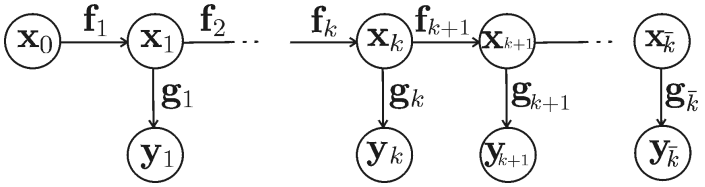

Figure 9: Graph associated with the autonomous system

4 Application to state estimation

4.1 Causal state estimator

To facilitate understanding, we shall consider the autonomous discrete-time system:

$$\left\{ \begin{array}{rcl} \mathbf{x}_k & = & \mathbf{f}_k(\mathbf{x}_{k-1}) \\ \mathbf{y}_k & = & \mathbf{g}_k(\mathbf{x}_k) \end{array} \right. \qquad k = 1, \ldots, \bar{k}, \tag{23}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the state vector and $\mathbf{y}_k \in \mathbb{R}^m$ is the output vector. It is a special case of the problem defined by (1), which could be treated in its general form along the same lines. The functions $\mathbf{f}_k$ and $\mathbf{g}_k$ may be nonlinear. At time k, the state estimator can use the measurement $\mathbf{y}_1^{\mathrm{m}}, \ldots, \mathbf{y}_k^{\mathrm{m}}$ in the causal case, and $\mathbf{y}_1^{\mathrm{m}}, \ldots, \mathbf{y}_k^{\mathrm{m}}, \mathbf{y}_{k+1}^{\mathrm{m}}, \ldots, \mathbf{y}_{\bar{k}}^{\mathrm{m}}$ in the noncausal case. The sets $\mathbb{X}_k$ and $\mathbb{Y}_k$ are deduced from prior information, or from the measurements $\mathbf{y}_k^{\mathrm{m}}$ via the interpretation function $\phi_{\mathbf{y}}$. In the absence of specific prior information, the prior sets $\mathbb{X}_0, \ldots, \mathbb{X}_{\bar{k}}$ are all taken as $\mathbb{R}^n$ and the prior sets $\mathbb{Y}_1, \ldots, \mathbb{Y}_{\bar{k}}$ are all taken as $\mathbb{R}^{\dim \mathbf{y}}$.

The SCSP $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$ associated with the state estimator is defined by

$$\begin{array}{l} \mathcal{V} = \{\mathbf{x}_0, \ldots, \mathbf{x}_{\bar{k}}, \mathbf{y}_1, \ldots, \mathbf{y}_{\bar{k}}\}, \\ \mathcal{D} = \{\mathbb{X}_0, \ldots, \mathbb{X}_{\bar{k}}, \mathbb{Y}_1, \ldots, \mathbb{Y}_{\bar{k}}\}, \\ \mathcal{C} = \big\{\mathbf{x}_\ell = \mathbf{f}_\ell(\mathbf{x}_{\ell-1});\ \ell = 1, \ldots, \bar{k}\big\} \cup \big\{\mathbf{y}_\ell = \mathbf{g}_\ell(\mathbf{x}_\ell);\ \ell = 1, \ldots, \bar{k}\big\}. \end{array} \tag{24}$$

Its graph is represented on Figure 9. Despite the presence of arrows, this graph is *not* oriented. The arrows are only meant to indicate the direction along which the associated function operates.

In the causal case, the generalized set estimator presented in Section 1 specializes into the following algorithm, where CSE stands for *Causal State Estimator*.

CSE

Input: $\mathbb{X}_0, \ldots, \mathbb{X}_{\bar{k}}, \mathbb{Y}_1, \ldots, \mathbb{Y}_{\bar{k}}$;

Initialization:

1 for $\ell := 1$ to $\bar{k}$, $\{\mathbb{X}_\ell := \mathbb{X}_\ell \cap \mathbf{f}_\ell(\mathbb{X}_{\ell-1}) \cap \mathbf{g}_\ell^{-1}(\mathbb{Y}_\ell);\}$;

2 for $\ell := \bar{k}$ down to 1, $\{\mathbb{Y}_\ell := \mathbf{g}_\ell(\mathbb{X}_\ell);\ \mathbb{X}_{\ell-1} := \mathbb{X}_{\ell-1} \cap \mathbf{f}_\ell^{-1}(\mathbb{X}_\ell);\}$;

3 $k := 1$;

Iteration k

4 wait for $\mathbf{y}_k^{\mathrm{m}}$;

5 $\mathbb{Y}_k := \mathbb{Y}_k \cap \phi_{\mathbf{y}}(\mathbf{y}_k^{\mathrm{m}})$;

6 $\mathbb{X}_k := \mathbb{X}_k \cap \mathbf{g}_k^{-1}(\mathbb{Y}_k)$;

7 for $\ell := k+1$ to $\bar{k}$, $\{\mathbb{X}_\ell := \mathbb{X}_\ell \cap \mathbf{f}_\ell(\mathbb{X}_{\ell-1});\ \mathbb{Y}_\ell := \mathbf{g}_\ell(\mathbb{X}_\ell);\}$;

8 for $\ell := k$ down to 1, $\{\mathbb{Y}_\ell := \mathbf{g}_\ell(\mathbb{X}_\ell);\ \mathbb{X}_{\ell-1} := \mathbb{X}_{\ell-1} \cap \mathbf{f}_\ell^{-1}(\mathbb{X}_\ell);\}$;

9 if $\big(k < \bar{k}\big)$ $\{k := k+1$; go to Step 4$\}$.

CSE is a specialization of the generalized set estimator presented in Section 1. It performs an optimal contraction during the initialization and after each measurement as stated by the following theorem.

Theorem 7 *After Step 2 and after each execution of Step 8 of* CSE*, $\mathcal{H}$ is minimal.* $\diamondsuit$

Proof: Step 1 corresponds to FALL($\mathcal{H}$), where $\mathcal{H} = (\mathcal{V}, \mathcal{D}, \mathcal{C}, \mathbf{x}_{\bar{k}})$ and Step 2 corresponds to CLIMB($\mathcal{H}$). From Theorem 6, these two steps produce the optimal contraction of $\mathcal{H}$. At Step 5, an external contraction of $\mathcal{H}$ takes place. After Step 5, $\mathcal{H}$ is thus no longer minimal, but it is still up-consistent, if we consider $\mathbf{y}_k$ as the new root of $\mathcal{H}$. Steps 6, 7 and 8 correspond to CLIMB($\mathcal{H}$). After Step 8, $\mathcal{H}$ is minimal from Theorem 5. $\diamondsuit$

In many practical situations, we are interested not in all variables but only in a few of them. In the context of state estimation, at time k, we may only want to estimate $\mathbf{x}_k$ and $\mathbf{y}_k$. Define the RCSE algorithm, for Recursive CSE, by replacing Steps 6, 7 and 8 in CSE by

6 $\mathbb{X}_k := \mathbb{X}_k \cap \mathbf{f}_k(\mathbb{X}_{k-1}) \cap \mathbf{g}_k^{-1}(\mathbb{Y}_k)$;

7 $\mathbb{Y}_k := \mathbb{Y}_k \cap \mathbf{g}_k(\mathbb{X}_k)$;

Note that Step 8 of CSE does not exist any more in RCSE. The following theorem shows that, although much simpler and much more efficient, RCSE provides the same accuracy for $\mathbf{x}_k$ and $\mathbf{y}_k$ as CSE.

Theorem 8 *After Step 2 and after Step 7 of* RCSE, $\mathbf{x}_k$ *and* $\mathbf{y}_k$ *are* $\mathcal{H}$*-consistent.* $\diamondsuit$

Proof: The consistency of $\mathbf{x}_k$ and $\mathbf{y}_k$ after Step 2 is a direct consequence of the minimality of $\mathcal{H}$ (see Theorem 7). Assume that Theorem 8 is true for $k-1$. Consider $\mathbf{x}_k$ as the root of $\mathcal{H}$. (i) Because $\mathcal{H}_{\mathbf{x}_{k-1}} = \text{SubTree}(\mathcal{H}, \mathbf{x}_{k-1})$ is up-consistent before Step 4, it remains so after Step 5. (ii) Because $\mathcal{H}_{\mathbf{x}_{k+1}} = \text{SubTree}(\mathcal{H}, \mathbf{x}_{k+1})$ is up-consistent after Step 2, it is also up-consistent after Step 5. (iii) Because $\mathcal{H}_{\mathbf{y}_k} = \text{SubTree}(\mathcal{H}, \mathbf{y}_{k-1})$ is a leaf, it is up-consistent. From (i), (ii) and (iii) and from Theorem 1, after Step 6 of RCSE, $\mathcal{H}$ is up-consistent and thus its root $\mathbf{x}_k$ is $\mathcal{H}$-consistent. From Theorem 3, $\mathbf{y}_k$ is $\mathcal{H}$-consistent after Step 7. $\diamondsuit$

Remark 3 RCSE *is similar to the algorithm proposed in [KJW98], [KJWM99]. The main difference is that in* RCSE *the initialization steps make it possible to take into account prior information on the future state or output vectors. Moreover, the use of the theory of set constraint satisfaction problems made it possible to derive Theorem 8 in a very simple way.* $\diamondsuit$

4.2 Non-causal state estimator

Consider now the noncausal case. Assume that the $\bar{k}$ domains $\mathbb{Y}_k, k \in \{1, \ldots, \bar{k}\}$ are available. The minimal contraction of $\mathcal{H}$ is computed using FALL-CLIMB, where the root is chosen as $\mathbf{x}_{\bar{k}}$. It can be translated, for this special problem, into the following *Non-Causal State Estimator* (NCSE) algorithm.

NCSE

Input: $\mathbb{X}_0, \ldots, \mathbb{X}_{\bar{k}}, \mathbb{Y}_1, \ldots, \mathbb{Y}_{\bar{k}}$;

Initialization:

1 for $k := 1$ to $\bar{k}$, $\mathbb{Y}_k := \mathbb{Y}_k \cap \phi_{\mathbf{y}}(\mathbf{y}_k^{\mathrm{m}})$;

2 for $k := 1$ to $\bar{k}$, $\{\mathbb{X}_k := \mathbb{X}_k \cap \mathbf{f}_k(\mathbb{X}_{k-1}) \cap \mathbf{g}_k^{-1}(\mathbb{Y}_k);\}$;

3 for $k := \bar{k}$ down to 1, $\{\mathbb{Y}_k := \mathbf{g}_k(\mathbb{X}_k);\ \mathbb{X}_{k-1} := \mathbb{X}_{k-1} \cap \mathbf{f}_k^{-1}(\mathbb{X}_k);\}$;

Step 2 corresponds to FALL and Step 3 corresponds to CLIMB. Note that the contracted domain $\mathbb{Y}_k$ after Step 1 contains the actual output vector $\mathbf{y}_k^*$ with a better accuracy than before Step 1.

4.3 Computer implementation

The computer implementation of FALL and CLIMB requires a representation for sets and an implementation of the local contraction operator $\rho_i()$. A set $\mathbb{V}$ is represented by a subpaving

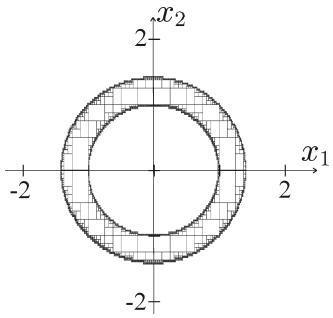

Figure 10: A subpaving (or union of boxes) is used to approximate a compact set

(union of boxes) that encloses it. For instance, the two-dimensional ring defined by $x_1^2 + x_2^2 \in [1, 2]$ can be represented by the subpaving of Figure 10.

In our context, the local contraction operator $\rho_i()$ corresponds either to the image $\mathbf{f}(\mathbb{V})$ of a set $\mathbb{V}$ by a vector function $\mathbf{f}$ or to the reciprocal image $\mathbf{f}^{-1}(\mathbb{W})$ of a set $\mathbb{W}$. The computation of a guaranteed enclosure of $\mathbf{f}(\mathbb{V})$ can be performed by the algorithm ImageSP [KJW98] and a guaranteed enclosure of $\mathbf{f}^{-1}(\mathbb{W})$ can be obtained with the algorithm Sivia [JW93]. These two algorithms are based on interval analysis [Moo79]. The following example illustrates the principle of ImageSP.

Example 7 *Consider again the set* $\mathbb{X} = \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1^2 + x_2^2 \in [1, 2]\}$ *and assume that* $\mathbf{f}(\mathbf{x}) = (x * y \ ; \ x + y)^T$. $\mathbb{X}$ *is approximated by the subpaving of Figure 11 (a). The first step performs a mincing of all boxes of this subpaving into smaller boxes as in Figure 11 (b). Then an outer approximation by a box of the image by* $\mathbf{f}$ *of each of these smaller boxes is computed using interval analysis. For instance, the image of the (much too large) box* $[\mathbf{x}] = [x_1] \times [x_2] = [1, 2] \times [3, 4]$ *would be approximated as follows:*

$$[\mathbf{y}] = ([x_1] * [x_2]) \times ([x_1] + [x_2]) = ([1, 2] * [3, 4]) \times ([1, 2] + [3, 4]) = [3, 8] \times [4, 6]. \qquad (25)$$

We thus obtain the union of all image boxes of Figure 11 (c). This union is guaranteed to contain $\mathbf{f}(\mathbb{X})$. *The last step of* ImageSP *is a reorganization of the boxes in order to get the subpaving of Figure 11 (d).* ♢

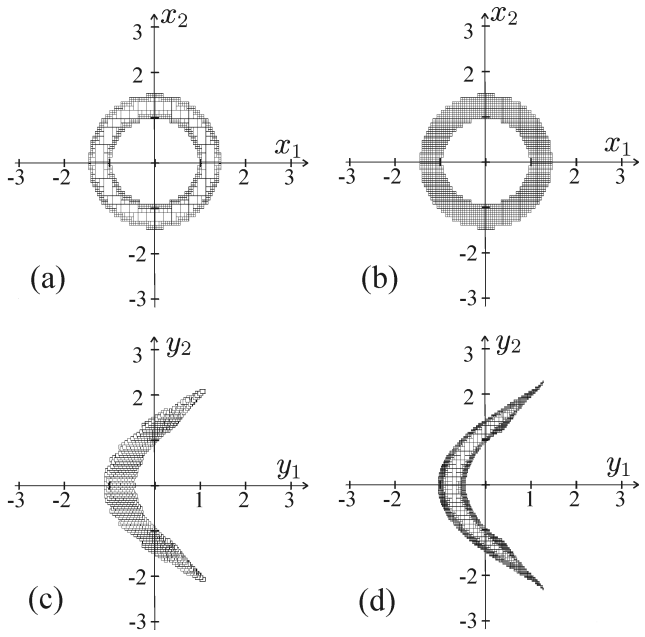

Figure 11: Principle of IMAGESP. (a) initial subpaving; (b) minced subpaving; (c) set approximating the image set, obtained by computing an outer approximation of the image by $\mathbf{f}$ of each of the boxes in (b); (d) final subpaving approximating the image set, obtained by simplifying (c)

5 Test case

Consider the nonlinear system

$$\left\{ \begin{array}{lcl} \left(\begin{array}{c} x_1(k) \\ x_2(k) \end{array} \right) & = & 3 \left(\begin{array}{c} \sin(x_1(k-1) + x_2(k-1)) \\ \cos(x_1(k-1) + x_2(k-1)) \end{array} \right) \\ y(k) & = & |x_1(k)| \end{array} \right. \quad \text{with } k \in \{1, \ldots, 10\}. \tag{26}$$

For $\mathbf{x}^*(0) = (0\ 0)^{\mathrm{T}}$, the values $\mathbf{x}^*(k)$ and $y^*(k), k \in \{1, \ldots, 10\}$, have been generated by simulation of this system. The measurements $y^{\mathrm{m}}(k)$ have then been obtained by adding a bounded noise to the actual output $y^*(k)$:

$$y^{\mathrm{m}}(k) = y^*(k) + n(k),$$

where $n(k)$ is a random variable with a uniform distribution in the interval $[-0.1, 0.1]$. The measurement uncertainty sets are taken as

$$\mathbb{Y}(k) = \phi_{\mathbf{y}(k)}(y^{\mathrm{m}}(k)) = y^{\mathrm{m}}(k) + [-0.1; 0.1]. \tag{27}$$

Note that the condition

$$y^*(k) \in \mathbb{Y}(k) \tag{28}$$

is satisfied for all k. The domains obtained for $\mathbf{x}(k)$ by RCSE and NCSE are depicted in Figure 12. The total computing time for RCSE and NCSE is less than one minute on a Pentium 133 MHz personal computer. The frames of all subfigures are $[-4, 4] \times [-4, 4]$. The initial domains given to the estimators are $\mathbb{X}(0) = \cdots = \mathbb{X}(10) = \mathbb{R}^2$ and $\mathbb{Y}(1) = \cdots = \mathbb{Y}(10) = \mathbb{R}$. The first subfigure is entirely grey, which means that RCSE is unable to provide any information about $\mathbf{x}(0)$ ($\mathbb{X}(0) = \mathbb{R}^2$), contrary to NCSE [3]. The last two subfigures, for $k = 10$, are identical because both estimators have now processed the same information.

6 Conclusions

Nonlinear parameter and state estimation has been cast into the general framework of SCSPs. The notion of SCSP is itself a generalization to the vector case of that of ICSP (Interval

[3] The lines obtained by the noncausal set estimator for $k = 0$ are due to the non-invertibility of $\mathbf{f}$. As can be seen from the picture for $k = 1$, $\mathbf{x}(1)$ is rather precisely estimated, with a value approximately equal to $(0; 3)^{\mathrm{T}}$. Solving $\mathbf{f}(\mathbf{x}(0))) = \mathbf{x}(1)$ amounts to solving $x_1(0) + x_2(0) = 2k\pi$, $k \in \mathbb{Z}$, the solutions of which correspond to a family of parallel lines in the $\mathbf{x}(0)$-space.

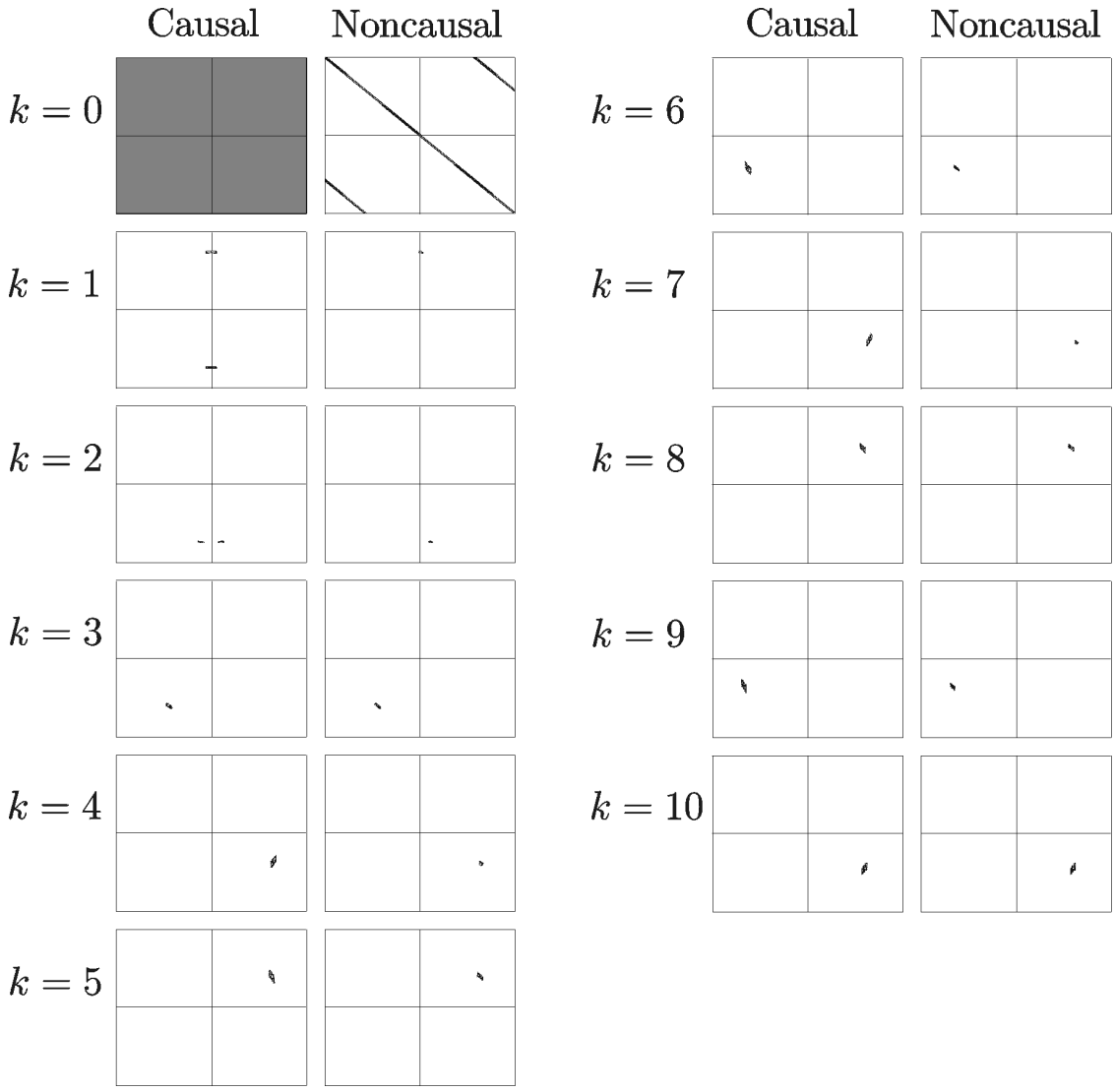

Figure 12: Causal and noncausal set estimations

Constraint Satisfaction Problem) well known in the area of artificial intelligence [Hyv92]. The main difference between SCSPs and ICSPs is that ICSPs handle intervals whereas SCSPs handle subsets of $\mathbb{R}^n$. This made it possible to obtain two powerful theorems (the propagation and retropropagation theorems) that are not true in an ICSP context. Existing algorithms [BGGP99], have been generalized to SCSPs whose graphs are trees, under the names of FALL and CLIMB. Contrary to what happens for ICSPs, we have shown that a single execution of FALL followed by a single execution of CLIMB was sufficient to produce an optimal contraction.

In the special case of causal state estimation, the algorithms proposed in [KJW98] and [KJWM99], can be interpreted in the framework of the more general FALL-CLIMB algorithm.

We have chosen in this paper not to put emphasis on computer implementation, but this is of course a critical issue, addressed in detail in [Kie99]. The source code in C++ Builder 4, and an executable program for IBM-compatible PCs corresponding to the example are available on request.

References

[AFL00] J. Arsouze, G. Ferrand, and A. Lallouet. Chaotic iterations for constraint propagation and labeling. Research Report RR-LIFO-2000-04, LIFO, Laboratoire d'Informatique Fondamentale d'Orléans, Université d'Orléans, BP 6759, F-45067 Orléans Cedex 2, France, 2000. Available at: ftp://ftp-lifo.univ-orleans.fr/pub/RR/RR2000/RR2000-04.ps.

[BGGP99] F. Benhamou, F. Goualard, L. Granvilliers, and J. F. Puget. Revising hull and box consistency. In *Proceedings of the International Conference on Logic Programming*, pages 230–244, Las Cruces, NM, 1999.

[Cle87] J. G. Cleary. Logical arithmetic. *Future Computing Systems*, 2(2):125–149, 1987.

[Dav87] E. Davis. Constraint propagation with interval labels. *Artificial Intelligence*, 32(3):281–331, 1987.

[Gel74] A. Gelb. *Applied Optimal Estimation.* MIT Press, Cambridge, MA, 1974.

[Hyv92] E. Hyvonen. Constraint reasoning based on interval arithmetics: the tolerance propagation approach. *Artificial Intelligence*, 58(1–3):71–112, December 1992.

[Jau00] L. Jaulin. Interval constraint propagation with application to bounded-error estimation. *Automatica*, 36(10):1547–1552, 2000.

[Jau01] L. Jaulin. Reliable minimax parameter estimation. *Reliable Computing*, 7(3):231–246, 2001.

[JW93] L. Jaulin and E. Walter. Set inversion via interval analysis for nonlinear bounded-error estimation. *Automatica*, 29(4):1053–1064, 1993.

[Kal60] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the AMSE, Part D, Journal of Basic Engineering*, 82:35–45, 1960.

[Kie99] M. Kieffer. *Estimation ensembliste par analyse par intervalles, application à la localisation d'un véhicule.* PhD dissertation, Université Paris-Sud, Orsay, France, 1999.

[KJW98] M. Kieffer, L. Jaulin, and E. Walter. Guaranteed recursive nonlinear state estimation using interval analysis. In *Proceedings of the 37th IEEE Conference on Decision and Control*, pages 3966–3971, Tampa, FL, 1998.

[KJWM99] M. Kieffer, L. Jaulin, E. Walter, and D. Meizel. Guaranteed mobile robot tracking using interval analysis. In *Proceedings of the MISC'99 Workshop on Applications of Interval Analysis to Systems and Control*, pages 347–359, Girona, Spain, 1999.

[KK98] W. Kang and A.J. Krener. Nonlinear observer design: A backstepping approach. In *Proceedings of the 13th International Symposium on the Mathematical Theory of Networks and Systems*, pages 245–248, Padova, Italy, 6-10 July 1998.

[KS67] M. G. Kendall and A. Stuart. *The Advanced Theory of Statistics, Vol. 2, Inference and Relationship, 2nd edition.* Charles Griffin and Company, London, 1967.

[Lan79] Y.D. Landau. *Adaptive Control, The Model Reference Approach.* Marcel Dekker, New York, NY, 1979.

[Lue66] D.G. Luenberger. Observers for multivariable systems. *IEEE Transactions on Automatic Control*, 11(2):190–197, 1966.

[MNPLW96] M. Milanese, J. Norton, H. Piet-Lahanier, and E. Walter, editors. *Bounding Approaches to System Identification.* Plenum Press, New York, NY, 1996.

[Moo79] R. E. Moore. *Methods and Applications of Interval Analysis.* SIAM, Philadelphia, PA, 1979.

[MR91] U. Montanari and F. Rossi. Constraint relaxation may be perfect. *Artificial Intelligence*, 48(2):143–170, 1991.

[MV91] M. Milanese and A. Vicino. Estimation theory for nonlinear models and set membership uncertainty. *Automatica*, 27(2):403–408, 1991.

[Nor94] J. P. Norton, editor. Special Issue on Bounded-Error Estimation: Issue 1. 1994. *International Journal of Adaptive Control and Signal Processing* **8**(1):1–118.

[Nor95] J. P. Norton, editor. Special Issue on Bounded-Error Estimation: Issue 2. 1995. *International Journal of Adaptive Control and Signal Processing* **9**(1):1–132.

[Sor83] H. Sorenson, editor. Special Issue on Applications of Kalman Filtering. 1983. *IEEE Transactions on Automatic Control* **28**(3):253–434.

[Wal75] 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. McGraw-Hill, New York, NY, 1975.

[Wal90] E. Walter, editor. Special Issue on Parameter Identification with Error Bounds. 1990. *Mathematics and Computers in Simulation* **32**(5-6):447–607.