23

Guaranteed Nonlinear Set Estimation via Interval Analysis

L. Jaulin and É. Walter

23.1. INTRODUCTION

Many methods have been developed for solving problems arising in mathematics and physics which are formulated in such a way as to require a point solution (e.g., a real number or vector). However, because of the uncertainty attached to the data and numerical errors induced by the finite-word-length representation in the computer, these methods are generally not appropriate to accurately characterize the uncertainty with which the solution is obtained. It is then difficult to assess the validity of the result.

Set formulation of problems replaces the search for a point solution by that of a feasible solution set that may contain a non-denumerable set of vectors. It is then possible to take uncertainty on the data as well as numerical errors into account and to get a global and guaranteed result. Uncertainty on this result can be computed rigorously, contrary to the classical point approaches. Interval analysis is one of the main tools that can be used to characterize sets obtained as the results of computations on sets. It generalizes real and vector calculi to intervals and vector intervals (or boxes). The manipulated subsets are approximated by sets consisting of unions of boxes (or subpavings). In set-inversion problems, which constitute a large part of set problems, the solution set is defined as the reciprocal image of a given set by

---

L. JAULIN AND É. WALTER • Laboratoire des Signaux et Systèmes, CNRS École Supérieure d'Electricité, 91192 Gif-sur-Yvette Cedex, France.

Bounding Approaches to System Identification, edited by M. Milanese *et al.* Plenum Press, New York, 1996.

a known function. The simple common structure of these problems makes it possible to derive a single algorithm that can be used to approximate the solution set for any set-inversion problem. This chapter applies this general approach to the problem of bounded-error estimation in the nonlinear case.

Let $M(.)$ be a set of models parameterized by a vector $\mathbf{p} \in \mathbb{R}^{n_p}$. To each value of $\mathbf{p}$ corresponds a model $M(\mathbf{p})$. Let $\mathbf{y} \in \mathbb{R}^{n_y}$ be the vector of all available experimental data, which may consist of measurements performed at various times. The corresponding model output will be denoted by $\mathbf{y}_m(\mathbf{p}) \in \mathbb{R}^{n_y}$. The dependency of $\mathbf{y}$ and $\mathbf{y}_m$ in the experimental conditions (inputs, measurement times, and so forth) is omitted to simplify notation. The output error is defined as

$$\mathbf{e}(\mathbf{p}) = \mathbf{y} - \mathbf{y}_m(\mathbf{p}). \tag{23.1}$$

Bounded-error estimation aims at characterizing the set $\mathbb{S}$ of all values of $\mathbf{p}$ such that $\mathbf{y}_m(\mathbf{p})$ is feasible in the sense that $\mathbf{e}(\mathbf{p})$ belongs to some prior feasible set for the errors $\mathbb{E}$. It is easy to deduce from $\mathbb{S}$ a point estimate $\hat{\mathbf{p}}$ for the parameters, as well as the uncertainty attached to it.

This chapter is organized as follows. Section 23.2 shows how bounded-error estimation can be formulated as a set-inversion problem and gives some illustrative test cases. The notions of interval analysis needed for the algorithm to be proposed are then presented in Section 23.3. Section 23.4 explains how pavings and subpavings can be used to approximate and bracket solution sets. Section 23.5 presents the set-inversion algorithm applied in Section 23.6 to the test cases presented in Section 23.2.

23.2. BOUNDED-ERROR ESTIMATION AS A SET-INVERSION PROBLEM

A MATLAB-like notation is used for vector equations and inequalities. Vectors and vector-valued functions are denoted by bold lower-case letters. Equalities and inequalities are to be understood componentwise. Note that some precautions are required in the manipulation of such operators. For instance, the contraposite of $\mathbf{u} \leq \mathbf{v}$ is not $\mathbf{u} > \mathbf{v}$ since the two proposals may be false simultaneously. Usual real functions such as sin, exp, and so forth, when their arguments are vectors, become vector functions and are also written in bold. They are evaluated component by component. For instance

$$\mathbf{sin}(\mathbf{u}) = \mathbf{sin}\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} \sin(u_1) \\ \sin(u_2) \\ \sin(u_3) \end{pmatrix}. \tag{23.2}$$

Let $\mathbf{f}$: $\mathbb{R}^n \rightarrow \mathbb{R}^p$ be a continuous function and $\mathbb{Y}$ be a closed subset of $\mathbb{R}^p$. Solving the associated set-inversion problem means characterizing

$$\mathbb{X} = \mathbf{f}^{-1}(\mathbb{Y}) = \{\mathbf{x} \mid \mathbf{f}(\mathbf{x}) \in \mathbb{Y}\}. \tag{23.3}$$

$\mathbb{X}$ is the solution set of the problem. The set function $\mathbf{f}^{-1}$ is the reciprocal function of $\mathbf{f}$. The direct image of $\mathbb{X}$ by $\mathbf{f}$ is defined by

$$\mathbf{f}(\mathbb{X}) = \{\mathbf{f}(\mathbf{x}) \mid \mathbf{x} \in \mathbb{X}\}. \tag{23.4}$$

The set $\mathbf{f}(\mathbb{R}^n)$ is a differential manifold, called the image manifold. When $p > n$, it is almost surely n-dimensional. Otherwise, it is almost surely p-dimensional. From elementary set theory, $\mathbf{f}(\mathbb{X}) \subset \mathbb{Y}$. In many practical problems, $\mathbb{Y}$ can be defined by a finite set of inequalities

$$\mathbb{Y} = \{\mathbf{y} \mid \mathbf{g}(\mathbf{y}) \leq \mathbf{0}\}. \tag{23.5}$$

The following equivalences then hold true

$$\mathbf{x} \in \mathbb{X} \Leftrightarrow \mathbf{f}(\mathbf{x}) \in \mathbb{Y} \Leftrightarrow \mathbf{g} \circ \mathbf{f}(\mathbf{x}) \leq \mathbf{0}. \tag{23.6}$$

If $\mathbf{h} = \mathbf{g} \circ \mathbf{f}$, $\mathbb{X}$ can be described by the finite set of inequalities

$$\mathbb{X} = \{\mathbf{x} \mid \mathbf{h}(\mathbf{x}) \leq \mathbf{0}\}. \tag{23.7}$$

Solving a set-inversion problem thus often amounts to characterizing a set defined by inequalities. When $\mathbf{h}$ is linear, $\mathbb{X}$ is a polyhedron, and its characteristics, such as its volume, the smallest box or ellipsoid containing it, can be computed accurately. When $\mathbf{h}$ is nonlinear, the techniques based on interval analysis presented in what follows make it possible to bracket $\mathbb{X}$ between simpler sets consisting of unions of boxes.

In the context of bounded-error estimation, the posterior feasible set for the parameters can be written as

$$\mathbb{S} = \{\mathbf{p} \mid \mathbf{e}(\mathbf{p}) \in \mathbb{E}\} = \mathbf{e}^{-1}(\mathbb{E}). \tag{23.8}$$

Characterizing $\mathbb{S}$ is, therefore, a problem of set inversion. The parameter vector $\mathbf{p}$, the error function $\mathbf{e}$ and the prior feasible set for the errors $\mathbb{E}$, respectively, stand for $\mathbf{x}$, $\mathbf{f}$ and $\mathbb{Y}$. In what follows, assume that $\mathbb{E}$ can be defined by a finite set of inequalities. Two test-cases are now introduced to illustrate the notions presented.

TEST-CASE 1: (Parameter estimation) Consider a two-parameter problem[1] where

$$\mathbf{y} = (0.1, 0.1)^{\mathrm{T}}. \tag{23.9}$$

These data correspond to two scalar measurements performed at times

$$\mathbf{t} = (0.5, 1)^{\mathrm{T}}. \tag{23.10}$$

The corresponding output for a model $M(\mathbf{p})$ is given by

$$\begin{aligned} \mathbf{y}_{\mathrm{m}}(\mathbf{p}) &= (0.5\cos(p_1) + 1.25)\,\mathbf{cos}(p_2\mathbf{t}) \\ &= \begin{pmatrix} (0.5\cos(p_1) + 1.25)\cos(p_2/2) \\ (0.5\cos(p_1) + 1.25)\cos(p_2) \end{pmatrix}, \end{aligned} \tag{23.11}$$

where the ith component of $\mathbf{y}_m(\mathbf{p})$ is computed for the ith component of $\mathbf{t}$. For $\mathbf{p}$ to be feasible, the error must satisfy

$$\mathbf{e}(\mathbf{p}) = \mathbf{y} - \mathbf{y}_m(\mathbf{p}) \in \mathbb{E} = \{\mathbf{e} \mid -\mathbf{0.75} \leq \mathbf{e} \leq \mathbf{0.75}\}, \tag{23.12}$$

where $\mathbf{0.75}$ is a vector with all entries equal to 0.75. The set to be characterized is given by $\mathbb{S} = \mathbf{e}^{-1}(\mathbb{E})$.

TEST-CASE 2: (State estimation) Consider the discrete-time state space model

$$\begin{cases} x_1(k+1) = \cos(x_1(k)\, x_2(k)) \\ x_2(k+1) = 3x_1(k) - \sin(x_2(k)) \\ y_m(k) = x_1^2(k) - x_2(k) \end{cases}, \qquad \mathbf{x}(0) = \begin{pmatrix} x_1(0) \\ x_2(0) \end{pmatrix} = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \mathbf{p}. \tag{23.13}$$

In order to estimate the unknown initial conditions, ten measurements $y(k)$ ($k = 0, \ldots, 9$) have been generated by simulating the model with $\mathbf{x}(0) = (2, 1)^{\mathrm{T}}$, which therefore correspond to the true value for the parameters. Adding a random error ε to each of these noise-free outputs, such that $-0.5 \leq \varepsilon \leq 0.5$, the resulting data set is then

$$\begin{aligned} \mathbf{y} &= (y(0), \ldots, y(9))^{\mathrm{T}} \\ &= (3, -5, 0.6, 2.2, -3.8, -1.4, 0.4, -1.2, -1.8, 2.6)^{\mathrm{T}}. \end{aligned} \tag{23.14}$$

The set $\mathbb{S}$ to be characterized is that of all $\mathbf{x}(0) = \mathbf{p}$ such that $\mathbf{e}(\mathbf{p}) \in \mathbb{E}$ with

$$\mathbb{E} = 0.5\,[-\mathbf{1}, \mathbf{1}], \tag{23.15}$$

where $[-\mathbf{1}, \mathbf{1}]$ stands for an axis-aligned hypercube centered on the origin and with width two. The error function $\mathbf{e}$ could be given a formal expression, but the result would be very complex. On the other hand, $\mathbf{e}$ is easily obtained by an algorithm. Using pseudo PASCAL, $\mathbf{e}(\mathbf{p})$ can be computed by

```
x1(0) := p1; x2(0) := p2;
For k := 0 to 9 do
  begin
    ym(k) := x1^2(k) - x2(k);
    e(k) := y(k) - ym(k);
    x1(k + 1) := cos(x1(k) * x2(k));
    x2(k + 1) := 3 x1(k) - sin(x2(k));
  end;
```

(23.16)

and

$$\mathbf{e}(\mathbf{p}) := \begin{pmatrix} e(0) \\ \cdots \\ e(9) \end{pmatrix}. \tag{23.17}$$

Again, the set to be characterized is $\mathbb{S} = \mathbf{e}^{-1}(\mathbb{E})$.

23.3. INTERVAL ANALYSIS

Interval calculus can be seen as a simple formalism for manipulating inequalities. In the interval approach to numerical computations,[2–4] any uncertain number is replaced by an interval guaranteed to contain it. Intervals are manipulated as a new type of numbers represented by an ordered pair of real numbers associated with the extremities of the interval. Intervals thus have a dual nature of numbers and infinite sets. Many algorithms take advantage of this duality and combine operations on sets, such as union and intersection, with arithmetical operations. High level languages implementing interval calculus are readily available.[5,6]

An interval $[x] \in \mathbb{R}$ or real interval is a closed, connected, and bounded subset of $\mathbb{R}$, such that

$$[x] = [x^-, x^+] = \{x \mid x^- \leq x \leq x^+\}. \tag{23.18}$$

The set of all real intervals will be denoted by $\mathbb{IR}$. Interval arithmetic generalizes addition, subtraction, multiplication, and division to intervals. If, for instance, $x^- \leq x \leq x^+, y^- \leq y \leq y^+$ and $z = x + y$, then $x^- + y^- \leq z \leq x^+ + y^+$ so that the addition of two intervals is defined as

$$[x] + [y] = \{x + y \mid x \in [x] \text{ and } y \in [y]\} = [x^- + y^-, x^+ + y^+], \tag{23.19}$$

Similarly,

$$-[x] = \{-x \mid x \in [x]\} = [-x^+, -x^-], \tag{23.20}$$

$$[x] - [y] = \{x - y \mid x \in [x] \text{ and } y \in [y]\} = [x^- - y^+, x^+ - y^-], \tag{23.21}$$

$$\text{If } 0 \notin [x], \text{ then } 1 / [x] = \{ 1 / x \mid x \in [x] \} = [1/x^+, 1/x^-], \tag{23.22}$$

$$[x] * [y] = [\min(x^- y^-, x^- y^+, x^+ y^-, x^+ y^+), \max(x^- y^-, x^- y^+, x^+ y^-, x^+ y^+)], \tag{23.23}$$

$$[x]^2 = \{x^2 \mid x \in [x]\}. \tag{23.24}$$

Note that $[x]^2 \neq [x] * [x]$. For instance, if $[x] = [-1, 1]$, then $[x]^2 = [0, 1]$ whereas $[x] * [x] = [-1, 1]$. It is easy to show that multiplication and addition are both associative and commutative. In general, however, addition is not distributive with respect to multiplication. The subdistributivity property guarantees that

$$[x] * ([y] + [z]) \subset [x] * [y] + [x] * [z]. \tag{23.25}$$

REMARK: When implementing interval arithmetic on a computer, one must take into account that not all intervals can be represented exactly and that approximations are committed at most arithmetical operations. It is necessary, therefore, to perform outwards rounding so as to insure that the exact results are contained in

the intervals computed. What follows does not consider these problems of implementation, which have little influence on the results for the problems treated.

The function $\mathbb{f}: \mathbb{IR} \to \mathbb{IR}$ is an inclusion function of the continuous function $f: \mathbb{R} \to \mathbb{R}$ if it satisfies

$$f([x]) \subset \mathbb{f}([x]) \tag{23.26}$$

for any $[x]$. Computing the interval $f([x])$ would require solving two global optimization problems, which is often exceedingly time consuming. On the other hand, for most real functions f, it is easy to obtain an inclusion function, as will be seen later. If the real x is known to belong to $[x]$, then $f(x)$ is guaranteed to belong to $\mathbb{f}([x])$. For any given f, there are, of course, infinitely many inclusion functions. One of them, denoted $[f]$, is minimal in the inclusion sense and satisfies $[f]([x]) = f([x])$ for any $[x]$. Call it the minimal inclusion function. For all elementary functions such as sin, cos, exp, log, arcsin, arccos, and so forth, this minimal inclusion function is easy to compute, as illustrated by the two following examples.

EXAMPLE 1: Since the exponential function is increasing, [exp] is given by

$$[\exp]([x]) = [\exp]([x^-, x^+]) = [\exp(x^-), \exp(x^+)] = \exp([x]). \tag{23.27}$$

To get the image interval, it therefore suffices to compute the image by exp of the extremities of $[x]$. This holds true for any real monotonic function.

EXAMPLE 2: Since the sine function is not monotonic, the technique of Example 1 cannot be applied to compute [sin]. It is easy, however, to show that [sin]([x]) can be computed as

$$\begin{aligned}
&\text{If } \exists\, k \in \mathbb{Z} \mid 2k\pi - \pi/2 \in [x] && \text{then } \sin^-([x]) := -1 \\
&&& \text{else } \sin^-([x]) := \min(\sin x^-, \sin x^+); \\
&\text{If } \exists\, k \in \mathbb{Z} \mid 2k\pi + \pi/2 \in [x] && \text{then } \sin^+([x]) := 1 \\
&&& \text{else } \sin^+([x]) := \max(\sin x^-, \sin x^+); \\
&[\sin]([x]) := [\sin^-([x]), \sin^+([x])]. &&
\end{aligned} \tag{23.28}$$

If f results from the composition of real operators or elementary functions, it is not possible to compute $[f]$. An inclusion function called *natural interval extension* can instead be obtained by replacing, in the formal expression for f, its argument x by the interval $[x]$ and the elementary functions and operators by the associated minimal inclusion functions. The natural interval extension is usually far from minimal, and may be much improved by suitably rewriting the formal expression of f or by taking advantage of the fact that the intersection of inclusion functions is an inclusion function.

EXAMPLE 3: Let $f_1(x) = x - x^2$ and $f_2(x) = x(1 - x)$. Although $f_1 \equiv f_2$, the two corresponding natural interval extensions $\mathbb{f}_1([x]) = [x] - [x]^2$ and $\mathbb{f}_2([x]) = [x] * (1 - [x])$ are different. The subdistributivity property of Eq. (23.25) implies $\mathbb{f}_2([x]) \subset$

$\mathbb{f}_1([x])$. For instance, if $[x] = [0, 1]$, $\mathbb{f}_1([x]) = [-1, 1]$ and $\mathbb{f}_2([x]) = [0, 1]$, when $f([x]) = [0, 1/4]$.

Vector calculus can similarly be extended to intervals by replacing vectors of $\mathbb{R}^n$ by boxes. A *box*, or *vector interval*, $[\mathbf{x}]$ of $\mathbb{R}^n$ consists of the Cartesian product of n scalar intervals. Boxes are indifferently denoted by

$$[\mathbf{x}] = [x_1^-, x_1^+] \times \ldots \times [x_n^-, x_n^+] = [x_1] \times \ldots \times [x_n] = [\mathbf{x}^-, \mathbf{x}^+] = \begin{pmatrix} [x_1^-, x_1^+] \\ \ldots \\ [x_n^-, x_n^+] \end{pmatrix}, \quad (23.29)$$

where $\mathbf{x}^- = (x_1^-, x_2^-, \ldots, x_n^-)^T$ and $\mathbf{x}^+ = (x_1^+, x_2^+, \ldots, x_n^+)^T$. The scalar intervals $[x_i] = [x_i^-, x_i^+]$ are the components of the box $[\mathbf{x}]$. The set of all boxes of $\mathbb{R}^n$ is denoted by $\mathbb{IR}^n$. The *width* of $[\mathbf{x}] \in \mathbb{IR}^n$ is given by

$$w([\mathbf{x}]) = \max_i \{x_i^+ - x_i^-\}. \quad (23.30)$$

When $w([\mathbf{x}]) = 0$, $[\mathbf{x}]$ degenerates into the vector $\mathbf{x}$, so that vectors can also be considered as belonging to $\mathbb{IR}^n$, with $\mathbf{x}^- = \mathbf{x}^+ = \mathbf{x}$. A *principal plane* of $[\mathbf{x}]$ is a symmetry plane of this box that is orthogonal to an axis i associated with a side of maximal length, i.e., $i \in \{j \mid w([\mathbf{x}]) = w([x_j])\}$.

Fig. 23.1 presents a two-dimensional box with its principal plane, a straight line here. The *enveloping box* $[\mathbb{A}]$ of a bounded set $\mathbb{A} \subset \mathbb{R}^n$ is the smallest box (in the sense of inclusion) of $\mathbb{IR}^n$ that contains $\mathbb{A}$.

$$[\mathbb{A}] = \cap \{[\mathbf{x}] \in \mathbb{IR}^n \mid \mathbb{A} \subset [\mathbf{x}]\}. \quad (23.31)$$

Vector addition and external multiplication can be extended to boxes:

$$[\mathbf{x}] + [\mathbf{y}] = \{\mathbf{x} + \mathbf{y} \mid \mathbf{x} \in [\mathbf{x}], \mathbf{y} \in [\mathbf{y}]\} = [\mathbf{x}^- + \mathbf{y}^-, \mathbf{x}^+ + \mathbf{y}^+], \quad (23.32)$$

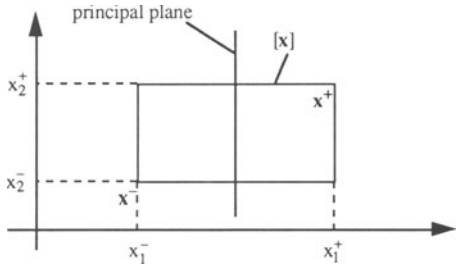


FIGURE 23.1. Box $[\mathbf{x}]$ with its principal plane.

$$\lambda[\mathbf{x}] = \{\lambda \mathbf{x} \mid \mathbf{x} \in [\mathbf{x}]\}. \tag{23.33}$$

The set function $\mathbb{f} : \mathbb{IR}^n \to \mathbb{IR}^p$ is an inclusion function of $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$ if and only if for any $[\mathbf{x}]$

$$\mathbf{f}([\mathbf{x}]) \subset \mathbb{f}([\mathbf{x}]). \tag{23.34}$$

It will be said to be *convergent* if for any sequence of boxes $[\mathbf{x}]$

$$w([\mathbf{x}]) \to 0 \Rightarrow w(\mathbb{f}([\mathbf{x}])) \to 0. \tag{23.35}$$

Convergent inclusion functions exist if and only if $\mathbf{f}$ is continuous. Among all possible inclusion functions $\mathbb{f}$ of $\mathbf{f}$: $\mathbb{R}^n \to \mathbb{R}^p$, one,

$$[\mathbf{f}]: \mathbb{IR}^n \to \mathbb{IR}^p; [\mathbf{x}] \to [\{\mathbf{f}(\mathbf{x} \mid \mathbf{x} \in [\mathbf{x}]\}], \tag{23.36}$$

is minimal in the sense of inclusion. Therefore, $[\mathbf{f}]([\mathbf{x}])$ is the enveloping box of the set $\mathbf{f}([\mathbf{x}])$. Fig. 23.2 illustrates the notion of inclusion function.

EXAMPLE 4: Consider the function

$$f: \mathbb{R}^4 \to \mathbb{R}; \mathbf{x} \to x_1 \exp(x_2) + x_3 \exp(x_4).$$

From the monotonicity of the exponential function,

$$\exp([x_i]) = [\exp]([x_i]) = [\exp(\underline{x}_i), \exp(\overline{x}_i)].$$

Therefore

$$[f]: \mathbb{IR}^4 \to \mathbb{IR}; [\mathbf{x}] \to [x_1] * \exp([x_2]) + [x_3] * \exp([x_4]).$$

Assume now that $x_1 > 0$ and $x_3 > 0$. Then

$$[f]([\mathbf{x}^-, \mathbf{x}^+]) = [\underline{x}_1 \exp(\underline{x}_2) + \underline{x}_3 \exp(\underline{x}_4), \overline{x}_1 \exp(\overline{x}_2) + \overline{x}_3 \exp(\overline{x}_4)].$$

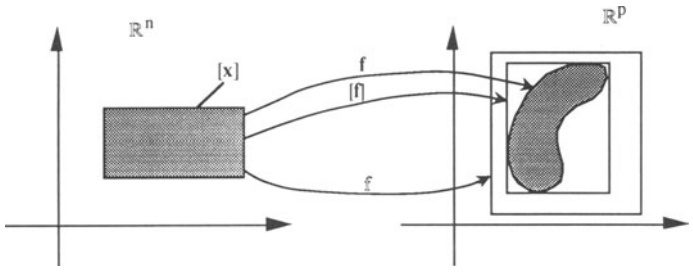


FIGURE 23.2. Minimal inclusion function $[\mathbf{f}]$ and inclusion function $\mathbb{f}$.

EXAMPLE 5: Consider the function f: $\mathbb{R}^2 \to \mathbb{R}$; $\mathbf{x} \to x_1 \sin x_2$. Since x_1 and x_2 appear independently in the formal expression of f, it is trivial to show that

$$[f]:\mathbb{I}\,\mathbb{R}^2 \to \mathbb{I}\,\mathbb{R};\ \mathbf{x} \to\ [x_1]*[\sin]([x_2]).$$

When **f** takes its values in $\mathbb{R}^p$, the coordinate functions of [**f**] are the minimal inclusion functions $[f_i]$ associated with the coordinate functions f_i of **f** ($i = 1, \ldots,$ p). As in the scalar case, a natural interval extension for **f** can be obtained by replacing in its formal expression (or in the algorithm describing **f**):

— the coordinates x_i of the argument **x** by the components $[x_i]$ of [**x**];
— all arithmetic operators by the corresponding operators for intervals; and
— all elementary functions by the corresponding minimal inclusion functions.

If each component of **x** appears at most once in the formal expression of a given coordinate function, then the natural interval extension is a minimal inclusion function.

TEST-CASE 1 (continued): The natural interval extension $\mathbb{e}([\mathbf{p}])$ for **e**(**p**) is given by

$$\begin{aligned}\mathbb{e}([\mathbf{p}]) &= \mathbf{y} - (0.5\ [\cos]([p_1]) + 1.25)\ [\mathbf{cos}]([p_2]\ \mathbf{t}) \\ &= \begin{pmatrix} 0.1 - (0.5\ [\cos]([p_1]) + 1.25)\ [\cos]([p_2]/2) \\ 0.1 - (0.5\ [\cos]([p_1]) + 1.25)\ [\cos]([p_2]) \end{pmatrix},\end{aligned} \tag{23.37}$$

where $[\cos]([x]) = [\sin](\pi/2 - [x])$ and [sin] is as in Example 2. Note that this inclusion function is minimal, so that $\mathbb{e}([\mathbf{p}]) = [\mathbf{e}]([\mathbf{p}]) = [\mathbf{e}([\mathbf{p}])]$.

TEST-CASE 2 (continued): The natural interval extension $\mathbb{e}([\mathbf{p}])$ for **e**(**p**) can be computed by the following pseudo-PASCAL code, where the inputs are $[p_1]$ and $[p_2]$

```
[x1] := [p1]; [x2] := [p2];
For k := 0 to 9 do
  begin
    [ym] := [x1]^2 - [x2];
    [e](k) := (y(k) - [ym])^2;
    [x1'] := cos([x1]*[x2]);
    [x2'] := 3 [x1] - sin([x2]);
    [x1] := [x1'];
    [x2] := [x2'];
  end;
```

(23.38)

The 10-dimensional box $\mathbb{e}([\mathbf{p}])$ whose kth component is given by $[e](k-1)$ is a convergent inclusion function for the error function **e**(**p**).

23.4. SET BRACKETING AND SUBPAVINGS

The solution set $\mathbb{S}$ to be characterized can usually be defined exactly, e.g., by the nonlinear inequalities of Eq. (23.7). However, the resulting description is often too complex to be of any use. It may, for instance, be difficult to know whether $\mathbb{S}$ is empty, whether it is connected, and what its volume or shape is. Another approach is to approximate $\mathbb{S}$ by more tractable sets, such as unions of boxes called subpavings. It will then become possible to approximate some characteristics of $\mathbb{S}$ by computing the corresponding characteristic of the approximating set.

A *subpaving* $\mathbb{K}$ of $\mathbb{R}^n$ is a set of non-overlapping boxes of $\mathbb{IR}^n$ with non-zero volume. If $\mathbb{A}$ is the subset of $\mathbb{R}^n$ consisting of the union of all boxes of $\mathbb{K}$, then $\mathbb{K}$ is a *paving* of $\mathbb{A}$. When there is no ambiguity, the set $\{\mathbb{K}\}$ consisting of the union of all boxes of $\mathbb{K}$ will also be denoted by $\mathbb{K}$. Subpavings are easily represented in a computer and readily amenable to set manipulation with the help of interval calculus. They are used to approximate, and more precisely bracket, the sets to be characterized. For almost any $\mathbb{X}$, it is possible to find two finite subpavings $\mathbb{X}^-$ and $\mathbb{X}^+$ such that $\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+$. The subpavings to be considered here always satisfy $\mathbb{X}^- \subset \mathbb{X}^+$ in the sense that each box of $\mathbb{X}^-$ is also a box of $\mathbb{X}^+$. The quantity $\Delta\mathbb{X} = \mathbb{X}^+ - \mathbb{X}^-$, therefore, is a subpaving, the *uncertainty layer*, which comprises all vectors for which it is not known whether they belong to the interior or exterior of $\mathbb{X}$. Fig. 23.3 illustrates the bracketing of a compact set between subpavings and the associated uncertainty layer. Let $V(\mathbb{X})$ be the set of all compacts $\mathbb{X}'$ such that $\mathbb{X}^- \subset \mathbb{X}' \subset \mathbb{X}^+$. In the Hausdorff distance sense, the diameter of $V(\mathbb{X})$ can be made as small as desired for almost any $\mathbb{X}$.[7] $V(\mathbb{X})$, therefore, is a neighborhood of $\mathbb{X}$.

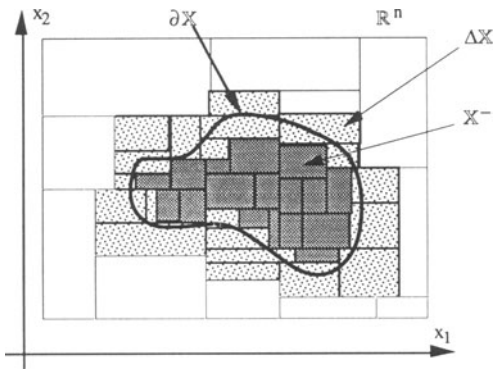


FIGURE 23.3. Bracketing a compact set between two subpavings.

23.5 SET INVERSION

To characterize $\mathbb{X} = f^{-1}(\mathbb{Y})$, assume that a convergent inclusion function $\mathbb{f}$ is known for f and that $\mathbb{Y}$ is compact. The notions of set inversion and bracketing of the solution set by subpavings are illustrated by Fig. 23.4. Note that

$$\mathbf{f}(\mathbb{X}) = \mathbf{f} \circ \mathbf{f}^{-1}(\mathbb{Y}) = \mathbb{Y} \cap \mathbf{f}(\mathbb{R}^n) \subset \mathbb{Y},$$

with $\mathbf{f}(\mathbb{X}) = \mathbb{Y}$ only if $\mathbf{f}$ is surjective, which is never true in the type of applications considered here. The algorithm *Set Inverter Via Interval Analysis* (SIVIA) will now be used to obtain the subpavings $\mathbb{X}^-$ and $\mathbb{X}^+$. It can also be used to bracket any quantity $Z(\mathbb{X})$ monotonic over $\mathbb{X}$ with as much precision as desired.[8] For simplicity, $\mathbb{X}$ is assumed to be bounded and included in a known prior box $[\mathbf{x}](0)$, which is used as the initial search domain. Extension to unbounded sets would involve the use of unbounded boxes (or generalized vector intervals).

A box $[\mathbf{x}]$ of $\mathbb{IR}^n$ is feasible if $[\mathbf{x}] \subset \mathbb{X}$, unfeasible if $[\mathbf{x}] \cap \mathbb{X} = \varnothing$, and ambiguous otherwise. Interval analysis provides two conditions, illustrated by Fig. 23.5, to test a box $[\mathbf{x}]$ for feasibility:

$$\mathbb{f}([\mathbf{x}]) \subset \mathbb{Y} \Rightarrow [\mathbf{x}] \subset \mathbb{X} \ ([\mathbf{x}] \text{ is feasible}), \tag{23.39}$$

$$\mathbb{f}([\mathbf{x}]) \cap \mathbb{Y} = \varnothing \Rightarrow [\mathbf{x}] \cap \mathbb{X} = \varnothing \ ([\mathbf{x}] \text{ is unfeasible}). \tag{23.40}$$

In all other cases, $[\mathbf{x}]$ is indeterminate. Note that indeterminate boxes may be feasible, unfeasible or ambiguous, but that any ambiguous box is indeterminate. Fig. 23.5 shows how an unfeasible box may be indeterminate, which explains why the two previous conditions are only sufficient.

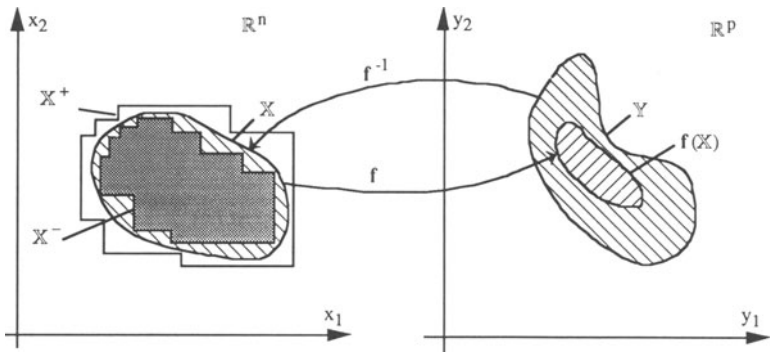


FIGURE 23.4. Bracketing the solution set of the set-inversion problem between two subpavings.

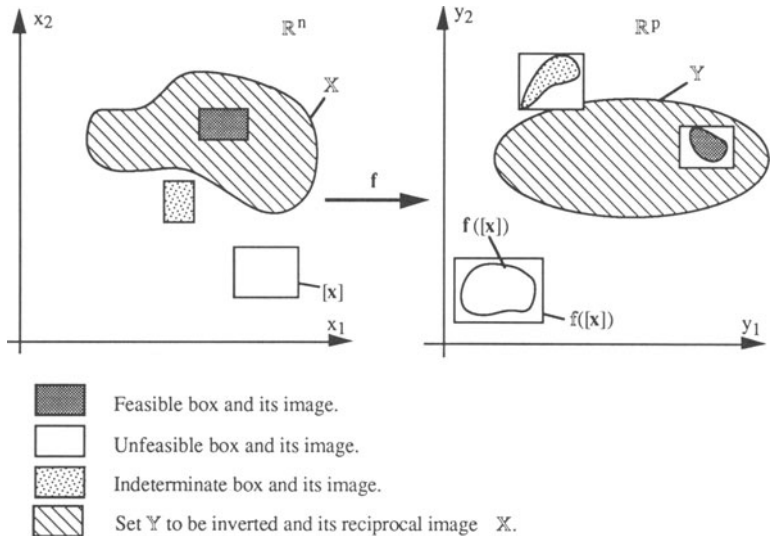


FIGURE 23.5. Sufficient conditions for a box to be feasible or unfeasible.

SIVIA involves three basic steps:

- the definition of a box of interest $[\mathbf{x}](0)$, on which the search will be performed;
- the choice of a paving $\mathbb{K}$ for $[\mathbf{x}](0)$; and
- the computation of $\mathbb{f}([\mathbf{x}])$ for each box of $\mathbb{K}$.

Three cases are then possible for any given box $[\mathbf{x}]$:

- if $\mathbb{f}([\mathbf{x}]) \subset \mathbb{Y}$ then $[\mathbf{x}] \subset \mathbb{X}$, ($[\mathbf{x}]$ is feasible);
- if $\mathbb{f}([\mathbf{x}]) \cap \mathbb{Y} = \emptyset$ then $[\mathbf{x}] \cap \mathbb{X} = \emptyset$, ($[\mathbf{x}]$ is unfeasible); and
- else, $[\mathbf{x}]$ is indeterminate.

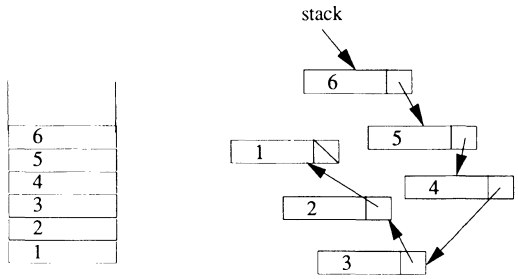
The paving $\mathbb{K}$ is thus partitioned into three subpavings $\mathbb{X}^-$, $\Delta\mathbb{X}$ et $\overline{\mathbb{X}^+}$, which correspond respectively to the sets of all feasible, indeterminate and unfeasible boxes. Since $\mathbb{X}^+ = \mathbb{X}^- \cup \Delta\mathbb{X}$,

$$\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+, \tag{23.41}$$

$$\partial\mathbb{X} \subset \Delta\mathbb{X}, \tag{23.42}$$

$$\text{vol}\,(\mathbb{X}^-) \leq \text{vol}\,(\mathbb{X}) \leq \text{vol}\,(\mathbb{X}^+), \tag{23.43}$$

$$[\mathbb{X}^-] \subset [\mathbb{X}] \subset [\mathbb{X}^+]. \tag{23.44}$$



(a) Conventional representation. (b) Computer representation.

FIGURE 23.6. Representation of a stack.

SIVIA recursively implements the idea of bracketing by subpavings that has just been presented. A *stack* of boxes (think of a stack of plates) is used, in which each element knows the location of the one located beneath it. Fig. 23.6 illustrates the representation of a stack on a computer. On this figure, a stack of six elements, numbered from 1 to 6, is presented under its traditional form (a) and as stored on the computer (b). Element 1 is at the bottom of the stack and Element 6 on top. The arrows represent the pointers, and the right-hand-side box of each element of the stack stands for the memory cell containing the address of the element beneath. Since it is at the bottom, Element 1 points to no other element. The topology of the stack is independent of its representation on the computer. Such a representation makes it possible to modify relationships without having to move the elements involved, which drastically speeds up the management of the memory.

Only three operations are possible on a stack, namely *stacking* (i.e., putting an element on top of the stack), *unstacking* (i.e., removing the element located on top of the stack) and testing the stack for emptiness. The box considered at iteration k is denoted by $[\mathbf{x}](k)$. The required accuracy for the subpavings $\mathbb{X}^-$ and $\Delta\mathbb{X}$ will be denoted by ε_r. After completion of the algorithm, all indeterminate boxes have a width smaller than or equal to ε_r. The inputs of SIVIA are the inclusion function $\mathbb{f}$, the set to be inverted $\mathbb{Y}$, the domain of interest $[\mathbf{x}](0)$ and the required accuracy ε_r. The initialization is performed by setting

$$[\mathbf{x}] = [\mathbf{x}](0), \text{ stack} := \varnothing, \mathbb{X}^- := \varnothing, \Delta\mathbb{X} := \varnothing, \tag{23.45}$$

and iteration is given by

Step 1	If $\mathbb{f}([\mathbf{x}]) \subset \mathbb{Y}$,	then $\mathbb{X}^- := \mathbb{X}^- \cup [\mathbf{x}]$. Go to Step 4.
Step 2	If $\mathbb{f}([\mathbf{x}]) \cap \mathbb{Y} = \varnothing$,	then go to Step 4.

Step 3 If $w([\mathbf{x}]) \leq \varepsilon_r$, then $\Delta\mathbb{X} := \Delta\mathbb{X} \cup [\mathbf{x}]$,
else bisect $[\mathbf{x}]$ along a principal plane and stack the two resulting boxes.

Step 4 If the stack is not empty, then unstack into $[\mathbf{x}]$ and go to Step 1.

End. (23.46)

SIVIA thus generates two subpavings $\mathbb{X}^-$ and $\Delta\mathbb{X}$. The dependency of these subpavings on ε_r is omitted to simplify notation. For almost any $\mathbb{X}$, the resulting bracketing ,

$$\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+ = \mathbb{X}^- \cup \Delta\mathbb{X} \tag{23.47}$$

defines a neighborhood of $\mathbb{X}$ with a diameter that tends to zero with ε_r. The convergence conditions are studied in Ref. 7. The main limitation of SIVIA lies in its computing time, which is proportional to the number of elements in $\mathbb{K}$ and increases exponentially with the number of parameters.[8]

When one is only interested in computing a characteristic of the solution set $\mathbb{X}$ such as its enveloping box $[\mathbb{X}]$ or its volume $\text{vol}(\mathbb{X})$, only the stack takes a significant place in memory. It is possible to avoid storing the subpavings $\mathbb{X}^-$ and $\Delta\mathbb{X}$ with the help of a recursive technique. Note, however, that the paving must be explored, even if it needs not be stored, so that the computing time is not shortened. To understand how one can avoid storing subpavings, let us modify SIVIA to recursively bracket the volume of $\mathbb{X}$. The program is initialized by setting stack := $\varnothing$, $\text{vol}^- := 0$, $\text{vol}^+ := 0$, $[\mathbf{x}] := [\mathbf{x}](0)$. The iteration is as follows

Step 1 If $\mathbb{f}([\mathbf{x}]) \subset \mathbb{Y}$, then $\text{vol}^- := \text{vol}^- + \text{vol}([\mathbf{x}])$,
$\text{vol}^+ := \text{vol}^+ + \text{vol}([\mathbf{x}])$, go to Step 4.

Step 2 If $\mathbb{f}([\mathbf{x}]) \cap \mathbb{Y} = \varnothing$, then go to Step 4.

Step 3 If $w([\mathbf{x}]) \leq \varepsilon_r$, then $\text{vol}^+ := \text{vol}^+ + \text{vol}([\mathbf{x}])$,
else bisect $[\mathbf{x}]$ along a principal plane and stack the two resulting boxes.

Step 4 If the stack is not empty, then unstack into $[\mathbf{x}]$ and go to Step 1.

End. (23.48)

As the volume is a monotonously increasing characteristic, completion gives

$$\text{vol}^- \leq \text{volume}(\mathbb{X}) \leq \text{vol}^+, \tag{23.49}$$

without having stored any subpaving. The transposition to the computation of the enveloping box $[\mathbb{X}]$ is trivial. The number of elements in the stack satisfies:[8]

$$\#\text{stack} < n \,\text{int}\,(\log_2(w([\mathbf{x}](0))) - \log_2(\varepsilon_r) + 1), \tag{23.50}$$

where int(r) stands for the integer part of a real r. Even for large n, the size of the stack remains reasonable. For instance if $n = 100$, $w([\mathbf{x}](0)) = 10^4$, and $\varepsilon_r = 10^{-10}$, then Eq. (23.50) implies that #stack < 4600.

SIVIA can easily be parallelized, and the following version can be implemented on r processors, which would similarly shorten the computing time. The inputs and initialization are as in the previous version of SIVIA. Iteration is as follows

Step 1 Split $[\mathbf{x}]$ into r boxes forming a subpaving $\mathbb{Z}$.

Step 2 Store in $\mathbb{X}^-$ all boxes $[\mathbf{z}]$ of $\mathbb{Z}$ such that $\mathbb{f}([\mathbf{z}]) \subset \mathbb{Y}$.

Step 3 Eliminate from $\mathbb{Z}$ all boxes $[\mathbf{z}]$ such that $\mathbb{f}([\mathbf{z}]) \cap \mathbb{Y} = \emptyset$.

Step 4 Store all boxes $[\mathbf{z}]$ such that $w([\mathbf{z}]) \leq \varepsilon_r$ in $\Delta\mathbb{X}$.

Step 5 Stack all remaining boxes of $\mathbb{Z}$.

Step 6 If the stack is not empty, unstack into $[\mathbf{x}]$ and go to Step 1.

End. (23.51)

Steps 2 to 5 are shared by all processors.

Frequently the parameter space is not isotropic because the sensitivities of $\mathbf{f}$ relative to the various components of $\mathbf{x}$ do not have the same order of magnitude. Bisecting along a principal plane, as suggested in the above description of SIVIA, may then turn out to be rather inefficient. The problem is to find a strategy for bisection to speed up the convergence. One way is to weight each component of $\mathbf{x}$ in such a way as to compensate for the anisotropy. It seems difficult, however, to suggest a rational strategy for the choice of the weights since the anisotropy may strongly depend on the position in the parameter space.

Another algorithm for set inversion was developed independently by Moore.[9] The main difference is that Moore's algorithm uses a queue when SIVIA uses a stack. The required memory for the queue is larger than for the stack by several orders of magnitude.

It may often be helpful to reformulate the problem of set inversion as that of finding any set $\hat{\mathbb{X}}$ such that

$$\mathbf{f}^{-1}(\mathbb{Y}^-) \subset \hat{\mathbb{X}} \subset \mathbf{f}^{-1}(\mathbb{Y}^+),$$

given two sets $\mathbb{Y}^-$ and $\mathbb{Y}^+$ such that $\mathbb{Y}^- \subset \mathbb{Y} \subset \mathbb{Y}^+$. The program is initialized by setting stack $:= \emptyset$, $\hat{\mathbb{X}} := \emptyset$, $[\mathbf{x}] := [\mathbf{x}](0)$. Iteration is as follows

Step 1 If $\mathbb{f}([\mathbf{x}]) \subset \mathbb{Y}^+$, then $\hat{\mathbb{X}} := \hat{\mathbb{X}} \cup [\mathbf{x}]$, go to Step 4.

Step 2 If $\mathbb{f}([\mathbf{x}]) \cap \mathbb{Y}^- = \emptyset$, then go to Step 4.

Step 3 Bisect $[\mathbf{x}]$ along a principal plane and stack the two resulting boxes.

Step 4 If the stack is not empty, then unstack into $[\mathbf{x}]$ and go to Step 1.

End. (23.52)

Even if $\mathbb{Y}$ is not known accurately, one then gets a characterization of the uncertainty attached to $\mathbf{x}$. Moreover, this algorithm does not require the specifica-

tion of ε_r. Provided that $\partial\mathbb{Y}^- \cap \partial\mathbb{Y}^+ = \varnothing$, $\hat{\hat{\mathbb{X}}}$ can be obtained in finite time. Even if $\mathbb{Y}$ is known exactly, such an approach remains of interest in the context of bounded-error estimation. By setting $\mathbb{Y}^- = \mathbb{E}$ and $\mathbb{X} = \mathbb{S}$, one gets $\mathbb{S} \subset \hat{\mathbb{S}}$, so that a set guaranteed to contain the posterior feasible set for the parameters is obtained. The set $\mathbb{Y}^+$ then plays the role of the stopping criterion. If the layer $\mathbb{Y}^+ - \mathbb{Y}^-$ is thick enough, this stopping criterion should make it possible to obtain a result comparable to that of SIVIA much more quickly. The set $\hat{\mathbb{S}}$ contains all the parameter vectors that are consistent with the available information, and none of those that are indisputably inconsistent (in the sense that their image is outside $\mathbb{Y}^+$).

23.6. EXAMPLES

TEST-CASE 1: For a required accuracy $\varepsilon_r = 0.04$ and a prior domain of interest $[\mathbf{p}](0) = [-10, 10] \times [-10, 10]$, SIVIA generates the paving presented on Fig. 23.7 in less than 160 seconds on a Compaq 386/33. It keeps less than 16 boxes in the stack at any given time (Eq. (23.50) predicts a number lower than or equal to 18).

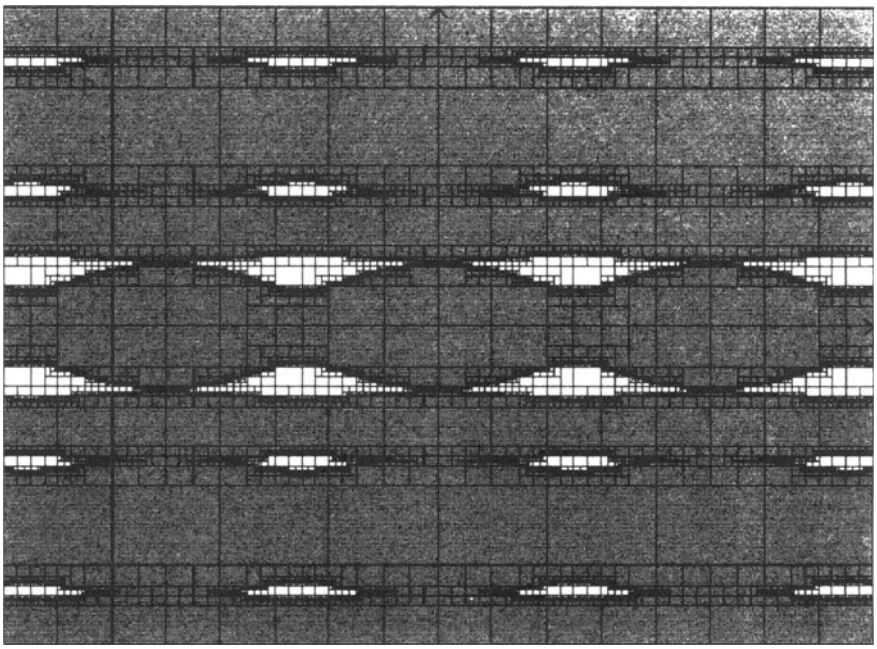


FIGURE 23.7. Paving generated by SIVIA for Test-case 1 in the (p_1, p_2) space. The frame corresponds to the search domain $[\mathbf{p}](0) = [-10, 10] \times [-10, 10]$.

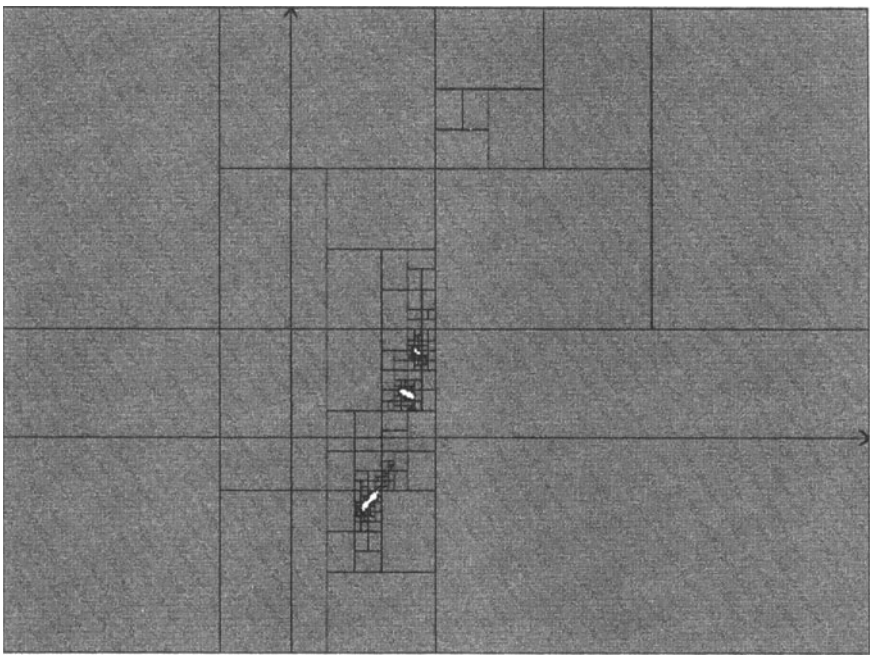


FIGURE 23.8. Paving generated by SIVIA for Test-case 2 in the (p_1, p_2) space. The frame corresponds to the search domain $[\mathbf{p}](0) = [-5, 10] \times [-5, 10]$.

The subpavings $\mathbb{S}^-$ and $\overline{\mathbb{S}^+}$ are filled in with white and grey, respectively. The volume of $\mathbb{S} \cap [\mathbf{p}](0)$ is guaranteed to satisfy

$$35 \leq \text{vol}(\mathbb{S} \cap [\mathbf{p}](0)) \leq 43. \tag{23.53}$$

The posterior feasible set $\mathbb{S}$ for the parameters turns out to be unconnected. One may wonder about the meaning of point estimation in this context.

TEST-CASE 2: For $\varepsilon_r = 0.05$ and $[\mathbf{p}](0) = [-5, 10] \times [-5, 10]$, the paving presented in Fig. 23.8 is generated in less than 10 seconds. The stack never contains more than 14 boxes ((Eq. 23.50) predicts a number lower than or equal to 18). The subpavings $\overline{\mathbb{S}^+}$ and $\Delta\mathbb{S}$ are filled in with grey and white, respectively. No box has been found in $\mathbb{S}^-$.

A random exploration of $[\mathbf{p}](0)$ with a uniform distribution for more than half an hour does not produce any feasible value for $\mathbf{p}$. To understand the difficulty of the problem, zoom around the true value for $\mathbf{p}$. For a required accuracy of $\varepsilon_r = 0.0001$, and $[\mathbf{p}](0) = [1.98, 2.02] \times [0.98, 1.02]$, SIVIA generates the paving presented in Fig. 23.9 in less than ten minutes. The subpavings $\mathbb{S}^-$ and $\overline{\mathbb{S}^+}$ are filled

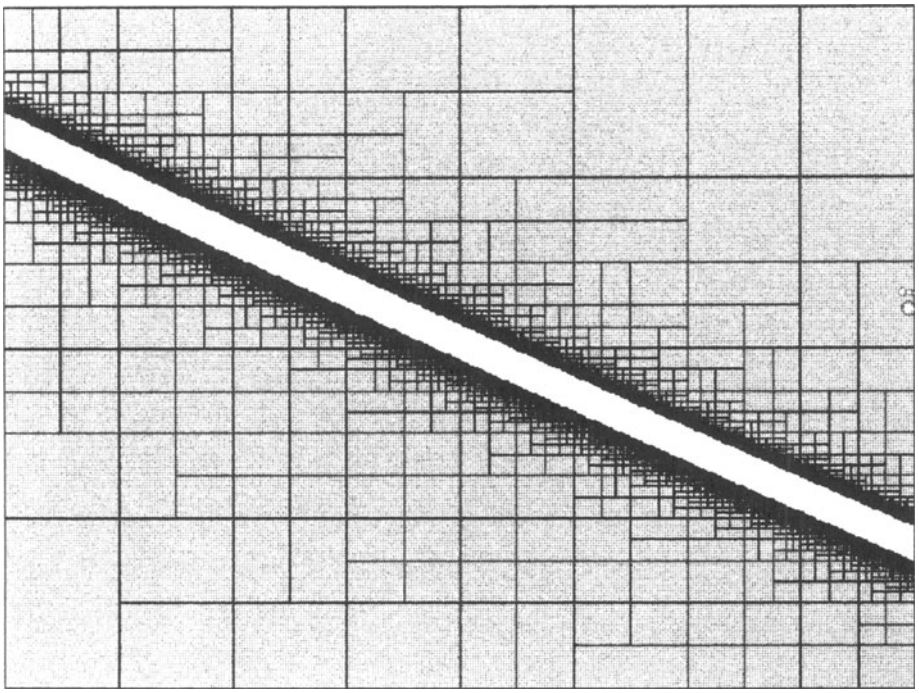


FIGURE 23.9. Paving generated by SIVIA for Test-case 2 in the (p_1, p_2) space. The frame corresponds to the search domain $[\mathbf{p}](0) = [1.98, 2.02] \times [0.98, 1.02]$.

in with white and grey, respectively. The posterior feasible set $\mathbb{S}$ is so narrow that it is almost impossible to reach it by random exploration.

23.7. CONCLUSIONS

Set inversion is particularly suitable to characterize the set of all values of parameters that are feasible in the sense that they satisfy a finite number of (possibly nonlinear) inequalities. The problem of estimating the parameters of a nonlinear model from bounded-error data is easily cast into this framework, which makes it possible to obtain approximate but guaranteed and global results in a finite number of operations.

The tools of interval analysis and the concept of subpaving have been used to derive efficient methods for the solution of the set-inversion problem. To the best of our knowledge, the only approach capable of providing guaranteed global results in nonlinear bounded-error estimation that is not based on interval analysis is the signomial approach advocated by Milanese and Vicino[10] and also presented in this

volume. Signomial analysis makes it possible to bracket the enveloping box $[\mathbb{S}]$ of $\mathbb{S}$ between two boxes. It gives $[\mathbb{S}]$ faster than SIVIA, which characterizes $\mathbb{S}$ in a much more detailed way and applies to a larger class of problems (e.g., problems involving trigonometric functions).

SIVIA eliminates a large portion of the domain of interest quickly before concentrating on the boundary of $\mathbb{S}$. It cannot provide great precision when there are more than a few parameters. However, it can still be useful provided that the required accuracy is suitably decreased. It may be, therefore, a powerful tool for a preliminary analysis before turning to local or random searches.

Among the many other problems that can be cast in the framework of set inversion, one may mention the problem of analyzing the robust stability of an uncertain time-invariant linear system.[11]

REFERENCES

1. L. Pronzato and E. Walter, *Math. Comput. Simul.* **32**, 571 (1990).
2. R. E. Moore, *Methods and Applications of Interval Analysis*, SIAM, Philadelphia, PA (1979).
3. A. Neumaier, *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge, United Kingdom (1990).
4. A. Ratschek and J. Rokne, *New Computer Methods for Global Optimization*, Ellis Horwood Limited, John Wiley & Sons, New York (1988).
5. IBM, *High-Accuracy Arithmetic Subroutine Library, (ACRITH): Program Description and User's Guide*, SC 33-6164-02, 3rd Ed. (1986).
6. R. Klate, U. W. Kulisch, M. Neaga, D. Ratz, and C. Ullrich, *PASCAL-XSC: Language Reference with Examples*, Springer-Verlag, Heidelberg, Germany (1992).
7. L. Jaulin and E. Walter, *Automatica* **29**, 1053 (1993).
8. L. Jaulin and E. Walter, *Math. Comput. Simul.* **35**, 123 (1993).
9. R. E. Moore, *Math. Comput. Simul.* **34**, 113 (1992).
10. M. Milanese and A. Vicino, *Automatica* **27**, 403 (1991).
11. E. Walter and L. Jaulin, *IEEE Trans. Autom. Control* **39**, 886 (1994).