

Exact recursive updating of uncertainty sets for discrete-time plants with a lag

Robin Hill¹, Yousong Luo² ^{*}

February 22, 2018

Abstract

In [7] there are new results concerning the polytopic set of possible states of a linear discrete-time SISO system subject to bounded disturbances from measurements corrupted by bounded noise. Using these results we construct an algorithm which, for the special case of a plant with a lag, recursively updates these polytopic sets when new measurements arrive.

1 INTRODUCTION

Determining the possible states of a dynamic system from noisy measurements, when the input and measurement errors are unknown except for bounds on their magnitude, is a computationally challenging task. We use the term *uncertainty set* to describe such a set of states consistent with past measurements and *a priori* assumptions on the exact plant model and bounds on the input and measurement noise.

The problem of recursively calculating uncertainty sets was formulated, and in principle solved, in the seminal papers [2, 16, 21]. There is an extensive literature on the topic of uncertainty set propagation. See [6] and [12] for background, the survey paper [11] and the book [3]. Some of the many other papers which consider this problem are [4], [14], [19] and [20]. In real-time applications, for example fault detection and isolation [15], existing exact algorithms are of limited use because of their computational complexity. For this reason there has been a lot of research recently on the use of zonotopes and constrained zonotopes to approximate the exact polytopic uncertainty set, for example [1, 5, 17].

In [7] two theorems have been presented from which new algorithms can be derived which update polytopic uncertainty sets exactly for linear, SISO, discrete-time plants. We believe that these algorithms run faster than any current exact schemes. The particular form that the new algorithms take depends on properties of the plant, and in this paper we present details for the special case of plants without a feedthrough term, that is plants with a lag. The theorems quoted from [7] cover both the lag and non-lag cases, but the details of algorithm development for a plant with a lag are a lot simpler than the non-lag case treated in [7].

Exact, recursive methods currently require polytopic projection techniques, for example Fourier-Motzkin elimination, see [8], [13] and [18]. In these papers it is the identification of redundant inequality constraints that is most demanding computationally. The algorithm we present relies on updating the vertex-facet incidence matrix and does not generate any redundant constraints. Another advantage is that, along with the incidence matrix, both the vertices and facets of the uncertainty set are updated. Existing methods update either the facets or vertices, but not both. It is well known that converting from a representation of the polytope in terms of inequality constraints to a vertex representation is not easy, so having both at our disposal is valuable.

We give representative plots which demonstrate that use our algorithm requires a smaller update time than two other commonly used methods.

2 Basic Setup

The material in this section and the next is a summary of results in [7], specialized to the case of a plant with no direct feedthrough. The plant P , a linear, time-invariant, causal discrete-time, m^{th} order scalar system, is assumed

^{*}¹Department of Electrical and Electronic Engineering, University of Melbourne, Melbourne, Vic 3010, Australia
robin.hill@unimelb.edu.au

[†]²School of Science, RMIT University, 124 Latrobe St, Melbourne, 3001, Australia yluo@rmit.edu.au

known. There are two sources of uncertainty, an input noise disturbance $(u_j)_{j=0}^\infty = \mathbf{u}$, and output measurement noise $(w_j)_{j=0}^\infty = \mathbf{w}$. The plant output is $(y_j)_{j=0}^\infty = \mathbf{y}$, and the measurement at time k is $z_k = y_k + w_k$. The initial state, at time $k = 0$, is assumed to be known exactly, but nothing is known about the uncertainties except that they satisfy $|u_j| \leq 1$ and $|w_j| \leq 1$.

Given an initial state \mathbf{x}_0 , the measurement history z_0, z_1, \dots, z_k , and the plant dynamics, we seek the uncertainty set at time $k + 1$, denoted S_{k+1} ; it is the set of possible states consistent with the measurements up to and including z_k , and is easily seen to be a closed, convex polytope.

2.1 Notation

The names of incidence vectors and matrices, whose elements are zeros or ones, will start with the letter \mathcal{I} . The λ -transform (generating function) of an arbitrary sequence $\mathbf{y} = (y_k)_{k=0}^\infty$ is defined to be $\hat{\mathbf{y}}(\lambda) := \sum_{k=0}^\infty y_k \lambda^k$. Real Euclidean space of dimension m is denoted \mathbb{R}^m , where m is the order of the plant P . States of the plant P are represented by vectors, or points, in \mathbb{R}^m .

2.2 Transfer function description

The plant for the estimation system has the transfer function representation $P(\lambda) = \hat{\mathbf{n}}(\lambda)/\hat{\mathbf{d}}(\lambda)$ where

$$\begin{aligned}\hat{\mathbf{n}}(\lambda) &= n_0 + n_1\lambda + n_2\lambda^2 + \dots + n_m\lambda^m \\ \hat{\mathbf{d}}(\lambda) &= d_0 + d_1\lambda + d_2\lambda^2 + \dots + d_m\lambda^m,\end{aligned}$$

$m \geq 1$, $\hat{\mathbf{n}}(\lambda)$ and $\hat{\mathbf{d}}(\lambda)$ are coprime polynomials with real coefficients, and it is assumed that both the plant $P(\lambda)$ and the plant $P^*(\lambda)$ for the dual system, defined below, are causal, implying $d_0 \neq 0$ and $d_m \neq 0$. It is assumed that the plant has a lag so $n_0 = 0$, and without loss of generality we take $d_0 = 1$.

Assuming zero initial conditions, \mathbf{y} and \mathbf{u} are related by $\hat{\mathbf{d}}(\lambda)\hat{\mathbf{y}}(\lambda) = \hat{\mathbf{n}}(\lambda)\hat{\mathbf{u}}(\lambda)$. The indeterminate λ represents the unit delay operator in the time domain.

2.3 State-space representations

A state-space description of the estimation system is

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}u_k \quad (1)$$

$$y_k = \mathbf{C}\mathbf{x}_k \quad (2)$$

$$z_k = y_k + w_k$$

where

$$\begin{aligned}\mathbf{A} &= \begin{bmatrix} \mathbf{0} & \mathbf{I}_{m-1} \\ -d_m & \dots & -d_1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}, \\ \mathbf{C} &= [n_m, \dots, n_1].\end{aligned} \quad (3a)$$

In (3a) \mathbf{I}_{m-1} denotes the $m - 1$ dimensional identity matrix, and $\mathbf{0}$ denotes a column vector of zeros of length $m - 1$. The system matrix \mathbf{A} is non-singular because $d_m \neq 0$.

There is a system closely related to the estimation system that we refer to as the dual system. Its input and output sequences are $(y_j^*)_{j=1}^\infty$ and $(u_j^*)_{j=1}^\infty$, and the dual plant, denoted P^* , has the transfer function representation

$$P^*(\lambda) = -\hat{\mathbf{n}}_{\text{dual}}(\lambda)/\hat{\mathbf{d}}_{\text{dual}}(\lambda) \quad (4)$$

where $\mathbf{n}_{\text{dual}} = (n_{m+1}, \dots, n_1)$ and $\mathbf{d}_{\text{dual}} = (d_{m+1}, \dots, d_2, 1)$. A minimal state-space realization of the dual system is

$$\mathbf{x}_{k+1}^* = \mathbf{A}^*\mathbf{x}_k^* + \mathbf{B}^*y_k^* \quad (5)$$

$$u_k^* = \mathbf{C}^*\mathbf{x}_k^* + D^*y_k^* \quad (6)$$

$$\mathbf{A}^* = \begin{bmatrix} -d_{m-1}/d_m & \mathbf{I}_{m-1} \\ \vdots & \\ -1/d_m & \mathbf{0} \end{bmatrix}, \quad (7)$$

$$\mathbf{B}^* = \begin{bmatrix} n_{m-1} \\ \vdots \\ n_0 \end{bmatrix} - \begin{bmatrix} d_{m-1} \\ \vdots \\ 1 \end{bmatrix} \frac{n_m}{d_m}, \quad (8)$$

$$\mathbf{C}^* = \begin{bmatrix} -1/d_m & 0 & \dots & 0 \end{bmatrix}, \quad D^* = \frac{-n_m}{d_m}. \quad (9)$$

2.4 Polytopes

The primal and dual states defined in the previous Section will be interpreted in terms of the geometry of the polytopic uncertainty set S_k , so in this Section we introduce notation and briefly summarise the relevant theory of convex polytopes. For more information and background, including the definition of a polytope, see for example [10] or [22]. Let S denote a closed, convex polytope.

The *support function* of S in \mathbb{R}^m is

$$h_S(\mathbf{f}) = \max_{\mathbf{x} \in S} \langle \mathbf{f}, \mathbf{x} \rangle,$$

where $\mathbf{f} \in \mathbb{R}^m$.

When $\mathbf{f} \neq \mathbf{0}$ the set

$$H_S(\mathbf{f}) := \{\mathbf{x} \in \mathbb{R}^m : \langle \mathbf{f}, \mathbf{x} \rangle = h_S(\mathbf{f})\}$$

is the supporting hyperplane of S with direction (outer normal vector) \mathbf{f} . If $\mathbf{f} = \mathbf{0}$ then $H_S(\mathbf{f}) = \mathbb{R}^m$.

The intersection of S with a supporting hyperplane is called a *face* of S , and a face of dimension $m-1$ is a *facet* of S . A face of dimension $m-2$ is called a *ridge*, and the faces of dimensions 0 and 1 are termed *vertices* and *edges*, respectively.

The *normal cone* at a boundary point \mathbf{x} is the set

$$\{\mathbf{f} \in \mathbb{R}^m : \langle \mathbf{f}, \mathbf{x} \rangle = h_S(\mathbf{f})\}$$

and is denoted $\mathcal{N}_S(\mathbf{x})$. It is generated by the outward normals to the facets that form the polytope at \mathbf{x} , that is

$$\mathcal{N}_S(\mathbf{x}) = \{\lambda_1 \mathbf{f}_1 + \dots + \lambda_n \mathbf{f}_n : \lambda_1, \dots, \lambda_n \geq 0\},$$

where $\mathbf{f}_1, \dots, \mathbf{f}_n$ are the directions of the facets containing \mathbf{x} . Thus $\mathcal{N}_S(\mathbf{x})$ contains the directions of all hyperplanes which touch S at \mathbf{x} .

The sets of vertices and boundary points of S are respectively denoted $\text{vert}(S)$ and ∂S . The interior of S is denoted $\text{int}(S)$. If $\mathbf{x} \in \text{int}(S)$, then $\mathcal{N}_S(\mathbf{x}) := \{\mathbf{0}\}$. By definition the directions of facets, and of the hyperplanes that contain facets, point outwards from the polytope. A direction is a non-zero vector.

The dual state $\mathbf{x}_k^*(\mathbf{y}^*, \mathbf{u}^*)$ can be interpreted as a direction vector \mathbf{f} in the normal cone of the primal state $\mathbf{x}_k(\mathbf{y}, \mathbf{u}) \in S_k$. We shall sometimes drop the subscript k , and indicate the next time instant with the superscript $+$. For example, \mathbf{f} will denote \mathbf{x}_k^* , and \mathbf{f}^+ will denote \mathbf{x}_{k+1}^* . Likewise, states \mathbf{x} and facets F belong to S_k , while \mathbf{x}^+ and F^+ belong to S_{k+1} .

2.4.1 Geometric and combinatorial description of S_k

Three matrices, \mathbf{V} , \mathbf{F} and \mathcal{I} are used to describe the polytope S_k , and the algorithm updates all of them. Geometric data are contained in \mathbf{V} and \mathbf{F} , while combinatorial information on the structure of S_k is contained in \mathcal{I} .

1. The m by n_v *vertex matrix* \mathbf{V} has columns containing the coordinates of the vertices of S_k .
2. The m by n_f *facet matrix* \mathbf{F} has columns containing the coordinates of the directions of the facets of S_k . Thus $\mathbf{F} = [\mathbf{f}_1, \dots, \mathbf{f}_{n_f}]$, where \mathbf{f}_i is a column vector representing the direction of facet number i . The magnitude of \mathbf{f}_i is not important, that is, for any $\alpha > 0$, $\alpha \mathbf{f}_i$ can be used in place of \mathbf{f}_i .
3. The *vertex-facet incidence matrix* of S_k is the matrix $\mathcal{I} \in \{0, 1\}^{n_f \times n_v}$ which has entry $\mathcal{I}(i, j) = 1$ if $\mathbf{v}_j \in F_i$, and $\mathcal{I}(i, j) = 0$ otherwise. So the facet F_i is the convex hull of the vertices identified by the ones in the i^{th} row of \mathcal{I} .

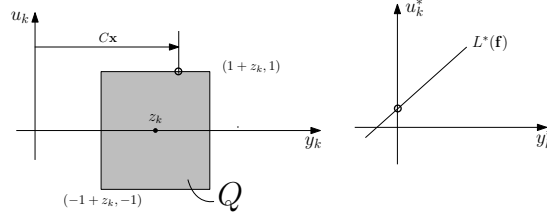


Figure 1: The circled points, at $(C\mathbf{x}, 1)$ in the $y_k u_k$ plane, and at the intersection of $L^*(\mathbf{f})$ and the u_k^* axis, are aligned, so $M(\mathbf{x}, \mathbf{f}, z_k) = \{(1, 0)\}$ because at the aligned points $u_k = 1$ and $y_k^* = 0$.

3 Statement of Required Theorems

Following Witsenhausen, [21], S_{k+1} is given recursively in terms of S_k and the new observation z_k by

$$S_{k+1} = \left\{ \begin{array}{l} \mathbf{x} \in S_k, \mathbf{x}^+ = \mathbf{A}\mathbf{x} + \mathbf{B}u_k, \\ \mathbf{x}^+ : y_k = C\mathbf{x}, \\ |u_k| \leq 1, |y_k - z_k| \leq 1. \end{array} \right\} \quad (10)$$

Special terminology is now introduced to describe states \mathbf{x} and \mathbf{x}^+ related as in (10).

Definition 1. The state $\mathbf{x} \in S_k$ is said to be a precursor of the state \mathbf{x}^+ , \mathbf{x} is propagated to \mathbf{x}^+ , and \mathbf{x}^+ is a successor to \mathbf{x} , if there exists a scalar u_k satisfying (1), (2), $|u_k| \leq 1$, and $|y_k - z_k| \leq 1$.

So S_{k+1} is the set of all successors to all states in S_k , and any precursor of any state $\mathbf{x}^+ \in S_{k+1}$ is in S_k .

We now define a relation between the inputs and outputs of the primal and dual systems.

Definition 2. The scalar pair (y_k, u_k) is said to be aligned with (y_k^*, u_k^*) if

$$\begin{aligned} u_k^* > 0 &\implies u_k = 1 \\ u_k^* < 0 &\implies u_k = -1 \\ |u_k| < 1 &\implies u_k^* = 0 \end{aligned} \quad (11)$$

and

$$\begin{aligned} y_k^* > 0 &\implies y_k = 1 + z_k \\ y_k^* < 0 &\implies y_k = -1 + z_k \\ |y_k - z_k| < 1 &\implies y_k^* = 0. \end{aligned} \quad (12)$$

Associated with any $\mathbf{f} \in \mathbb{R}^m$ define in the $y_k^* u_k^*$ -plane the dual line $L^*(\mathbf{f})$:

Definition 3. $L^*(\mathbf{f}) = \{(y_k^*, u_k^*) : n_m y_k^* + d_m u_k^* = -(\mathbf{f})_1\}$.

The scalars y_k^* and u_k^* , the input and output of the dual plant at time k , are constrained to lie on the line $L^*(\mathbf{f})$ by (6) and (9).

The square Q in the $y_k u_k$ -plane contains points (y_k, u_k) satisfying $|u_k| \leq 1$ and $|y_k - z_k| \leq 1$.

One more definition is required in order to state the Theorems quoted from [7].

Definition 4. Given $\mathbf{x} \in S_k$, $\mathbf{f} \in \mathcal{N}_{S_k}(\mathbf{x})$ and z_k , the set $M(\mathbf{x}, \mathbf{f}, z_k)$ is the set of scalar pairs (u_k, y_k^*) which satisfy

1. $(y_k, u_k) \in Q$, where $y_k = C\mathbf{x}$ and
2. (y_k, u_k) is aligned with (y_k^*, u_k^*) , where $(y_k^*, u_k^*) \in L^*(\mathbf{f})$.

Finding M is computationally very simple. For example, in Fig. 1 alignment occurs solely between the two circled points, so M contains the singleton pair $(u_k, y_k^*) = (1, 0)$.

We now present the Theorems from which the algorithm is derived. See Fig. 2 for a geometric depiction of the vectors in Theorem 5 for the important special case where \mathbf{x} and \mathbf{x}^+ are both boundary points. Proofs are in [7].

Theorem 5. Suppose $\mathbf{x} \in S_k$ and $\mathbf{f} \in \mathcal{N}_{S_k}(\mathbf{x})$. Then $\mathbf{x}^+ = \mathbf{A}\mathbf{x} + \mathbf{B}u_k \in S_{k+1}$ and $\mathbf{f}^+ = \mathbf{A}^*\mathbf{f} + \mathbf{B}^*y_k^* \in \mathcal{N}_{S_{k+1}}(\mathbf{x}^+)$ if and only if $(u_k, y_k^*) \in M(\mathbf{x}, \mathbf{f}, z_k)$.

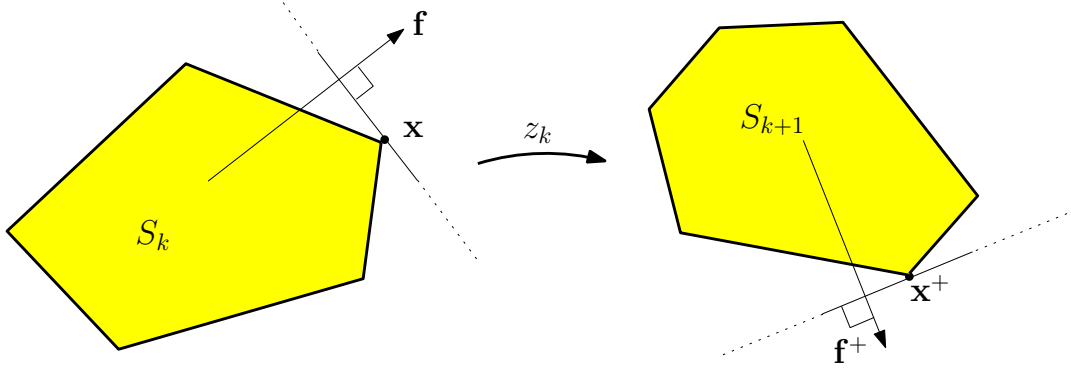


Figure 2: The vectors in Theorem 5. The state $\mathbf{x} \in \partial S_k$ and direction $\mathbf{f} \in \mathcal{N}_{S_k}(\mathbf{x})$ are propagated to $\mathbf{x}^+ \in \partial S_{k+1}$ and $\mathbf{f}^+ \in \mathcal{N}_{S_{k+1}}(\mathbf{x}^+)$ by the measurement z_k .

In a typical application a point on the boundary of S_k , and the direction of any one of its supporting hyperplanes, are propagated to a point on the boundary of S_{k+1} along with the direction of one of its supporting hyperplanes.

The companion result Theorem 6, given below, shows that any boundary point $\mathbf{x}^+ \in S_{k+1}$, and any direction in the normal cone of \mathbf{x}^+ , are attainable from *any* precursor \mathbf{x} of \mathbf{x}^+ and *some* direction in the normal cone of \mathbf{x} .

Theorem 6. *Select any $\mathbf{x}^+ \in S_{k+1}$, any $\mathbf{f}^+ \in \mathcal{N}_{S_{k+1}}(\mathbf{x}^+)$ and any precursor \mathbf{x} of \mathbf{x}^+ . There exists $\mathbf{f} \in \mathcal{N}_{S_k}(\mathbf{x})$ and $(u_k, y_k^*) \in M(\mathbf{x}, \mathbf{f}, z_k)$ for which $\mathbf{x}^+ = \mathbf{A}\mathbf{x} + \mathbf{B}u_k$ and $\mathbf{f}^+ = \mathbf{A}^*\mathbf{f} + \mathbf{B}^*y_k^*$.*

In the rest of the paper we show how to use the propagation of states and their normal cones to update the uncertainty set for plants with a lag. An implementation of our algorithm in Matlab code, `uncertaintysetwithlag.m`, is available on the link below¹. Also given is code `tcomplag.m` which allows comparisons of computation time for our algorithm, denoted F-V (facets-vertices), Fourier-Motzkin and mplp (multi-parametric linear programming). We used the software package [9] for Fourier-Motzkin projection and mplp. See [18] for an explanation of the use of Fourier-Motzkin projection to compute updated uncertainty sets.

4 The incidence matrix for S_k

We turn now to the construction of \mathcal{I}^+ , the incidence matrix for S_{k+1} , shown in Table 1, where the six incidence matrices in the body of the Table are defined in Section 5. Initially \mathcal{I}^+ is put equal to the $(3n_f + n_R)$ by $2n_v$ zero matrix, that is \mathcal{I}^+ is allocated sufficient space to accommodate all conceivable facets and vertices of S_{k+1} . So there are $3n_f + n_R$ rows in the Table, where n_f and n_R are respectively the number of facets and ridges in S_k . The rows are divided into four blocks, each of the first three blocks comprising n_f rows, and the last block n_R rows. The first two blocks point to facets of S_{k+1} whose directions have the property $(\mathbf{f})_m \neq 0$, while the last two blocks point to facets whose directions \mathbf{f} have the property $(\mathbf{f})_m = 0$.

Turning now to the columns of Table 1, the first block of n_v columns points to states $\mathbf{A}\mathbf{v}_j + \mathbf{B}$, $\mathbf{v}_l \in \text{vert}(S_k)$, which, depending on z_k , might be vertices of S_{k+1} . The second block of n_v columns points to states $\mathbf{A}\mathbf{v}_j - \mathbf{B}$, also possible vertices of S_{k+1} .

After \mathcal{I}^+ has been constructed in accordance with Table 1 it must have some rows and columns composed entirely of zeros. The final step is for these rows and columns of zeros to be deleted.

5 Filling in Table 1

We introduce notation to describe two ways a state can be propagated from S_k to S_{k+1} . The mappings g^T and g^B , where the superscripts indicate the top and bottom sides of Q , map a state on the boundary of S_k to the boundary of S_{k+1} .

On the top (bottom) side of Q , $u_k = 1$ ($u_k = -1$), and (1) is used to propagate \mathbf{x} .

¹Go to <https://mathworks.com/matlabcentral/fileexchange>

Facet directions \ Vertices	$g^T(\mathbf{V})$	$g^B(\mathbf{V})$
$A^*\mathbf{f}_i$ (with $u_k = 1$)	\mathcal{I}^T	$\mathbf{0}$
$A^*\mathbf{f}_i$ (with $u_k = -1$)	$\mathbf{0}$	\mathcal{I}^B
$A^*\mathbf{f}_i$ (where $(\mathbf{f})_1 = 0$)	$\mathcal{I}\mathcal{O}^T$	$\mathcal{I}\mathcal{O}^B$
$A^*\mathbf{f}_l^R$ (Facets of S_{k+1} from ridges)	$\mathcal{I}R^T$	$\mathcal{I}R^B$

Table 1: The incidence matrix for S_{k+1} prior to the removal of rows and columns composed entirely of zeros

Definition 7. For any $\mathbf{x} \in S_k$ put

$$\begin{aligned} g^T(\mathbf{x}) &= \mathbf{A}\mathbf{x} + \mathbf{B} \\ g^B(\mathbf{x}) &= \mathbf{A}\mathbf{x} - \mathbf{B}. \end{aligned} \quad (13)$$

Definition 8. A point $\mathbf{x} \in S_k$ is said to propagate up (resp. down) if $g^T(\mathbf{x})$ ($g^B(\mathbf{x})$) lies on the boundary of S_{k+1} .

These propagations occur when the maximum or minimum allowable value of the input (± 1) is applied to the plant, and furthermore the resulting state of the plant lies on the boundary of S_{k+1} . Theorem 5 tells us, given \mathbf{x} and $\mathcal{N}_{S_k}(\mathbf{x})$, if \mathbf{x} propagates up or down. For example, in Fig. 1, if $\mathbf{x} \in \partial S_k$ then \mathbf{x} propagates up to $g^T(\mathbf{x})$ because there exists $\mathbf{0} \neq \mathbf{f} \in \mathcal{N}_{S_k}(\mathbf{x})$ which implies $(u_k, y_k^*) = (1, 0) \in M(\mathbf{x}, \mathbf{f}, z_k)$ and so, by Theorem 5, $\mathbf{x}^+ = \mathbf{A}\mathbf{x} + \mathbf{B} \in S_{k+1}$ and $\mathbf{f}^+ = \mathbf{A}^*\mathbf{f} \in \mathcal{N}_{S_{k+1}}(\mathbf{x}^+)$ and is non-zero, further implying that $\mathbf{x}^+ \in \partial S_{k+1}$.

Definition 8 can be extended to facets as follows.

Definition 9. Suppose F is a facet S_k , and F^+ is a facet S_{k+1} . Then F is said to propagate up (down) to F^+ if $F^+ = g^T(F)$, ($F^+ = g^B(F)$).

Here $g^T(F)$, for example, denotes the set $\{g^T(\mathbf{x}) : \mathbf{x} \in F\}$.

If F propagates either up or down to F^+ then F^+ is an affinely isomorphic image of F because \mathbf{A} is invertible. Let F_i be any facet of S_k , and denote its direction by \mathbf{f}_i . There are three possibilities:

1. The line $L^*(\mathbf{f}_i)$ intersects the positive u_k^* axis, that is $d_m(\mathbf{f}_i)_1 < 0$. The $n_f \times 1$ logical vector $\mathcal{I}F^T$ points to facets with such directions. Explicitly, $\mathcal{I}F^T(i) = 1$ if \mathbf{f}_i satisfies $d_m(\mathbf{f}_i)_1 < 0$, and $\mathcal{I}F^T(i) = 0$ otherwise;
2. $L^*(\mathbf{f}_i)$ intersects the negative u_k^* axis, that is $d_m(\mathbf{f}_i)_1 > 0$. Define $\mathcal{I}F^B(i) = 1$ if \mathbf{f}_i satisfies $d_m(\mathbf{f}_i)_1 > 0$, and $\mathcal{I}F^B(i) = 0$ otherwise; and
3. $L^*(\mathbf{f}_i)$ passes through the origin. Define $\mathcal{I}F^O(i) = 1$ if \mathbf{f}_i satisfies $(\mathbf{f}_i)_1 = 0$, and $\mathcal{I}F^O(i) = 0$ otherwise.

5.0.1 Case 1

For every \mathbf{x}_j in F_i we have the situation depicted in Fig. 1. By Theorem 5, \mathbf{x}_j is propagated up, and F_i is propagated up to the facet $F_i^+ := g^T(F_i)$ of S_{k+1} , which has direction $\mathbf{A}^*\mathbf{f}_i$. The logical matrix $\mathcal{I}^T \in \{0, 1\}^{n_f \times n_v}$ is defined to have entry $\mathcal{I}^T(i, j) = 1$ if $\mathcal{I}(i, j) = 1$ and $\mathcal{I}F^T(i) = 1$, and $\mathcal{I}^T(i, j) = 0$ otherwise. A one in the i^{th} row and j^{th} column of \mathcal{I}^T implies that there is a facet, F_i^+ , of S_{k+1} with direction $\mathbf{A}^*\mathbf{f}_i$, and that $g^T(\mathbf{v}_j) \in F_i^+$.

Case 2) is similar to Case 1), except that F_i is propagated down. The matrix $\mathcal{I}^B \in \{0, 1\}^{n_f \times n_v}$ is defined to have entry $\mathcal{I}^B(i, j) = 1$ if $\mathcal{I}(i, j) = 1$ and $\mathcal{I}F^B(i) = 1$, and $\mathcal{I}^B(i, j) = 0$ otherwise.

We require for Case 3), covered below, as well as for the case of propagation of facets from ridges, to be treated in the next section, two logical matrices which point to those vertices of S_k that are propagated to vertices of S_{k+1} . The $n_v \times 1$ vector $\mathcal{I}V^{\text{PT}}$ is defined by $\mathcal{I}V^{\text{PT}}(j) = 1$ if the j^{th} column of \mathcal{I}^T is not composed entirely of zeros, and $\mathcal{I}V^{\text{PT}}(j) = 0$ otherwise. Thus $\mathcal{I}V^{\text{PT}}$ points to those vertices which propagate up to a vertex of S_{k+1} . Likewise $\mathcal{I}V^{\text{PB}}$ is defined by $\mathcal{I}V^{\text{PB}}(j) = 1$ if the j^{th} column of \mathcal{I}^B is not composed entirely of zeros, and $\mathcal{I}V^{\text{PB}}(j) = 0$ otherwise.

Every point of Q is aligned with the origin in the $y_k^* u_k^*$ plane so, in Case 3), by Theorem 5 every \mathbf{x}_j in F_i is propagated to $\mathbf{x}_j^+ = \mathbf{A}\mathbf{x}_j + \mathbf{B}u_k \in \partial S_{k+1}$ for all $|u_k| \leq 1$. We show the method for constructing all vertices of the

facet F_i^+ , and provide a sketch of the proof that it works. Put $F_i^+ = \{\mathbf{x}_j^+ : \mathbf{x}_j^+ = \mathbf{A}\mathbf{x}_j + \mathbf{B}u_k \text{ where } \mathbf{x}_j \in F_i, u_k \in [-1, 1]\}$. A convexity argument shows

$$F_i^+ = \{\mathbf{x}_j^+ : \mathbf{x}_j^+ = g^T(\mathbf{v}_j) \text{ or } \mathbf{x}_j^+ = g^B(\mathbf{v}_j), \mathbf{v}_j \in \text{vert}(F_i)\}.$$

We seek the vertices of the facet F_i^+ . Now every precursor \mathbf{v}_j of any \mathbf{x}_j^+ in the vertex set of F_i^+ is a vertex of F_i . However, it is not necessarily the case that every successor of every vertex of F_i is a vertex of F_i^+ . We must remove from F_i^+ those \mathbf{x}_j^+ which are not vertices of F_i^+ , and this can be done by removing those \mathbf{x}_j^+ which are not vertices of S_k .

Define the incidence matrix $\mathcal{IO}^T(i, j) = 1$ if $\mathcal{IF}^O(i) = 1$, $\mathcal{IV}^{\text{PT}}(j) = 1$ and $\mathcal{I}(i, j) = 1$, and $\mathcal{IO}^T(i, j) = 0$ otherwise. In words, $\mathcal{IO}^T(i, j) = 1$ if \mathbf{v}_j propagates up to a vertex of S_k , and \mathbf{v}_j is in F_i , with $(\mathbf{f}_i)_1 = 0$.

The matrix \mathcal{IO}^B , pointing to those vertices of facets with direction vector whose first component is zero that propagate down, is defined similarly. Thus $\mathcal{IO}^B(i, j) = 1$ if $\mathcal{IF}^O(i) = 1$, $\mathcal{IV}^{\text{PB}}(j) = 1$ and $\mathcal{I}(i, j) = 1$, and $\mathcal{IO}^B(i, j) = 0$ otherwise.

This completes the construction of all of Table 1 except the last block.

5.1 Propagation from a ridge in S_k to a facet in S_{k+1}

In the previous Section Case 3) was more intricate than Cases 1) and 2) because it treated facets with directions whose dual lines passed through the origin, opening up an infinite number of alignment possibilities. There is one, and only one, other situation in which a boundary point \mathbf{x} of S_k can have a direction \mathbf{f} in its normal cone which satisfies $(\mathbf{f})_1 = 0$. If such \mathbf{x} does not lie on a facet covered by case 3) above, \mathbf{x} must belong to a special type of ridge, to be defined next.

See Fig. 3 where S_k possesses two intersecting facets, with directions \mathbf{f}_1 and \mathbf{f}_2 , whose first components have opposite signs. These facets have the state \mathbf{x} in their intersection, so necessarily there exists a direction in the normal cone of \mathbf{x} whose first component is zero. We define R to be the set of all such ridges of S_k , and denote its cardinality by n_R . Clearly R is non-empty if S_k is non-empty and full-dimensional.

Definition 10. Define R to be the set of ridges, R_l , $l = 1, \dots, n_R$, of S_k with the property that R_l is the intersection of two facets F_1 and F_2 say, for which $(\mathbf{f}_1)_1 < 0$ and $(\mathbf{f}_2)_1 > 0$, where \mathbf{f}_1 and \mathbf{f}_2 are the directions of F_1 and F_2 .

Associated with any ridge in R there is by definition a direction in the cone generated by \mathbf{f}_1 and \mathbf{f}_2 having the property that its first component is zero. This direction is unique.

Definition 11. For any $R_l \in R$ define \mathbf{f}_l^R to be the direction in the convex hull of \mathbf{f}_1 and \mathbf{f}_2 for which $(\mathbf{f}_l^R)_1 = 0$.

Clearly $\mathbf{f}_l^R \in \mathcal{N}_{S_k}(\mathbf{x})$ for all $\mathbf{x} \in R_l$. The proof of the following Proposition is omitted.

Proposition 12. For all $R_l \in R$ there is a facet of S_k with direction $\mathbf{A}^*\mathbf{f}_l^R$.

For any S_k and any $R_l \in R$, the direction \mathbf{f}_l^R can be found using linear interpolation. More precisely $\mathbf{f}_l^R = t\mathbf{f}_1 + (1-t)\mathbf{f}_2$ where $t = (\mathbf{f}_1)_1 / [(\mathbf{f}_2)_1 - (\mathbf{f}_1)_1]$, and \mathbf{f}_1 and \mathbf{f}_2 are the directions of the facets which intersect to form R_l .

The incidence matrices in the bottom row of Table 1 can now be defined.

The vertex set of R_l is the intersection of the vertex sets of the facets which intersect to form R_l . The matrix $\mathcal{IRV} \in \{0, 1\}^{n_R \times n_v}$ has entry $\mathcal{IRV}(l, j) = 1$ if the ridge R_l contains \mathbf{v}_j , and $\mathcal{IRV}(l, j) = 0$ otherwise. So the l^{th} row of \mathcal{IRV} points to the vertex set of R_l .

The matrix $\mathcal{IR}^T \in \{0, 1\}^{n_R \times n_v}$ has entry $\mathcal{IR}^T(l, j) = 1$ if $\mathcal{IV}^{\text{PT}}(j) = 1$ and $\mathcal{IRV}(l, j) = 1$, and $\mathcal{IR}^T(l, j) = 0$ otherwise. So the l^{th} row of \mathcal{IR}^T points to vertices \mathbf{v}_j of R_l which propagate up to vertices of the ridge R_l in S_k .

The matrix $\mathcal{IR}^B \in \{0, 1\}^{n_R \times n_v}$ has entry $\mathcal{IR}^B(l, j) = 1$ if $\mathcal{IV}^{\text{PB}}(j) = 1$ and $\mathcal{IRV}(l, j) = 1$, and $\mathcal{IR}^B(l, j) = 0$ otherwise. So the l^{th} row of \mathcal{IR}^B points to vertices \mathbf{v}_j of R_l which propagate down to vertices of the ridge R_l in S_k .

This completes the definition of all of the incidence matrices occurring in Table 1. We claim that \mathcal{I}^+ constructed according to Table 1 is, after removal of rows and columns containing all zeros, the incidence matrix of S_{k+1} . We have sketched the proof, which ultimately relies on Theorems 5 and 6.

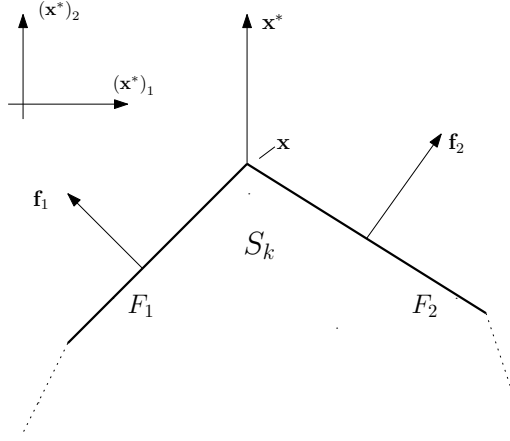


Figure 3: Two facets intersect in a ridge containing \mathbf{x} . The direction $\mathbf{x}^* \in \mathcal{N}_{S_k}(\mathbf{x})$ satisfies $(\mathbf{x}^*)_1 = 0$.

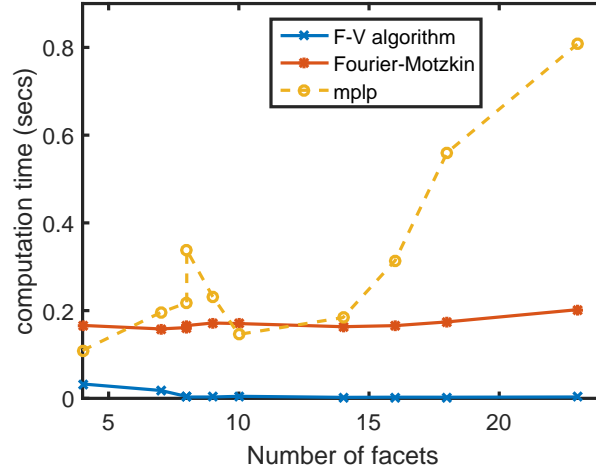


Figure 4: The time taken to update S_k versus the number of facets of S_k for a third order plant with a lag.

6 Examples

Theorems 5 and 6 are most naturally interpreted as yielding S_{k+1} , consistent with measurements up to z_k . The input is S_k consistent with measurements up to z_{k-1} . In common terminology we “update then propagate”. It is more common to determine S_{k+1} consistent with measurements up to z_{k+1} , with input S_k consistent with measurements up to z_k , essentially “propagate then update”. The accompanying code implements this latter case. For plants with a lag changing the order of the operations update and propagate is very easy.

We give representative plots comparing our algorithm, denoted F-V, projection using Fourier-Motzkin elimination, and projection using multi-parametric linear programming. They were generated using the Matlab code accompanying the paper. The reader can generate plots for her/himself with differing plant sizes and lengths of measurement sequences. Figure 4 compares the computation times of the three methods for a third order randomly selected stable plant, with randomly selected measurements. Figure 5 compares the computation times for a fifth order randomly selected stable plant, again with randomly chosen measurements. It can be seen that the algorithm presented in this paper is significantly faster. For these examples the F-V algorithm was between twenty and forty times faster, on average, than the better of the two other methods.

7 CONCLUSIONS

For the case of plants with a lag we have described a new method of uncertainty set propagation that appears to be more efficient than those in current use. There are in addition other advantages. It provides more information

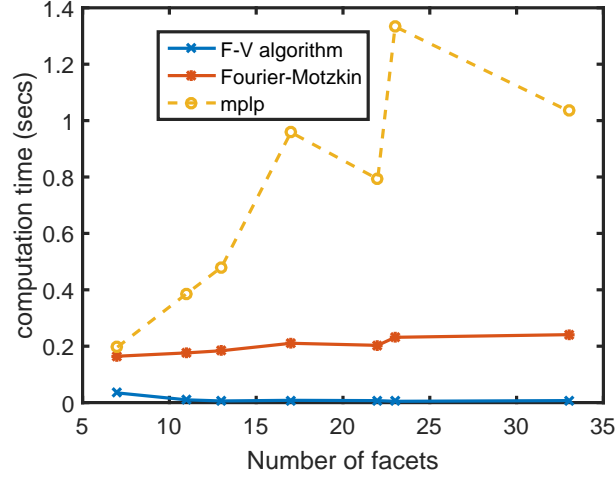


Figure 5: The time taken to update S_k versus the number of facets of S_k for a fifth order plant with a lag.

than most current methods, because the vertices and facets are both updated at every time step. Finally, because the computationally intensive steps are done using Boolean matrices, for moderately sized problems all calculations can be performed exactly over the rationals.

References

- [1] T. ALAMO, J.M. BRAVO, M.J. REDONDO, AND E.F. CAMACHO, *A set-membership state estimation algorithm based on dc programming*, Automatica, 44 (2008), pp. 216 – 224.
- [2] DIMITRI P. BERTSEKAS AND IAN B. RHODES, *Recursive state estimation for a set-membership description of uncertainty*, IEEE Trans. Automatic Control, AC-16 (1971), pp. 117–128.
- [3] FRANCO BLANCHINI AND STEFANO BIANI, *Set-Theoretic Methods in Control*, Birkhauser, Boston, 2008.
- [4] FRANCO BLANCHINI AND MARIO SZNAIER, *A convex optimization approach to synthesizing bounded complexity ℓ^∞ filters*, IEEE Trans. Automat. Control, 57 (2012), pp. 216–221.
- [5] CHRISTOPHE COMBASTEL, *Zonotopes and kalman observers: Gain optimality under distinct uncertainty paradigms and robust convergence*, Automatica, 55 (2015), pp. 265 – 273.
- [6] ELI FOGEL AND Y.F. HUANG, *On the value of information in system identification: Bounded noise case*, Automatica, 18 (1982), pp. 229 – 238.
- [7] ROBIN HILL, YOUSONG LUO, AND UWE SCHWERTDFEGER, *Exact recursive updating of uncertainty sets*, arxiv.org/abs/1612.04918v5, also under review, (2017).
- [8] S. S. KEERTHI AND E.G. GILBERT, *Computation of minimum-time feedback control laws for discrete-time systems with state-control constraints*, Automatic Control, IEEE Transactions on, 32 (1987), pp. 432–435.
- [9] M. KVASNICA, P. GRIEDER, AND M. BAOTIĆ, *Multi-Parametric Toolbox (MPT)*, 2004.
- [10] STEVEN R. LAY, *Convex sets and their applications*, Dover, Mineola, New York, 2007.
- [11] M. MILANESE AND A. VICINO, *Optimal estimation theory for dynamic systems with set membership uncertainty: An overview*, Automatica, 27 (1991), pp. 997 – 1009.
- [12] BRETT NINNESS AND GRAHAM C. GOODWIN, *Estimation of model quality*, Automatica, 31 (1995), pp. 1771 – 1797. Trends in System Identification.
- [13] S. V. RAKOVIC AND D. Q. MAYNE, *State estimation for piecewise affine, discrete time systems with bounded disturbances*, in Proc. of the 43rd. Conf. on Decision and Control, Bahamas, December 2004, pp. 3557–3562.

- [14] PAULO ROSA, CARLOS SILVESTRE, AND MICHAEL ATHANS, *Model falsification using set-valued observers for a class of discrete-time dynamic systems: a coprime factorization approach*, Int. J. Robust. Nonlinear Control, 24 (2014), pp. 2928–2942.
- [15] PAULO ROSA, CARLOS SILVESTRE, JEFF S. SHAMMA, AND MICHAEL ATHANS, *Fault detection and isolation of ltv systems using set-valued observers*, in Proc. of the 49th. Conf. on Decision and Control, Atlanta, December 2010, pp. 768–773.
- [16] F.C. SCHWEPPE, *Recursive state estimation: Unknown but bounded errors and system inputs*, Automatic Control, IEEE Transactions on, 13 (1968), pp. 22–28.
- [17] JOSEPH K. SCOTT, DAVIDE M. RAIMONDO, GIUSEPPE ROBERTO MARSEGLIA, AND RICHARD D. BRAATZ, *Constrained zonotopes: A new tool for set-based estimation and fault detection*, Automatica, 69 (2016), pp. 126 – 136.
- [18] J.S. SHAMMA AND KUANG-YANG TU, *Set-valued observers and optimal disturbance rejection*, Automatic Control, IEEE Transactions on, 44 (1999), pp. 253–264.
- [19] A.A. STOORVOGEL, *l_1 state estimation for linear systems using nonlinear observers*, in Decision and Control, 1996., Proceedings of the 35th IEEE Conference on, vol. 3, 1996, pp. 2407–2411 vol.3.
- [20] ROBERTO TEMPO, *Robust estimation and filtering in the presence of bounded noise*, IEEE Trans. Automat. Control, 33 (1988), pp. 864–867.
- [21] H. S. WITSENHAUSEN, *Sets of possible states of linear systems given perturbed observations.*, IEEE Trans. Automatic Control, AC-13 (1968), pp. 556–558.
- [22] GUNTER M. ZIEGLER, *Lectures on Polytopes*, Springer, New York, 1995.