Sparsity in Max-Plus Algebra and Systems

Anastasios Tsiamis * and Petros Maragos †

January 31, 2018

Abstract

We study sparsity in the max-plus algebraic setting. We seek both exact and approximate solutions of the max-plus linear equation with minimum cardinality of support. In the former case, the sparsest solution problem is shown to be equivalent to the minimum set cover problem and, thus, NP-complete. In the latter one, the approximation is quantified by the \( \ell_1 \) residual error norm, which is shown to have supermodular properties under some convex constraints, called lateness constraints. Thus, greedy approximation algorithms of polynomial complexity can be employed for both problems with guaranteed bounds of approximation. We also study the sparse recovery problem and present conditions, under which, the sparsest exact solution solves it. Through multi-machine interactive processes, we describe how the present framework could be applied to two practical discrete event systems problems: resource optimization and sparsity-aware system identification. Finally, we study a numerical example of the sparsity-aware system identification problem.

1 Introduction

Max-plus algebra has been used to model a subclass of nonlinear phenomena with some linear-like structure. It is obtained from the linear algebra if we replace addition with maximum and multiplication with addition (Butkovic, 2010). The development of this algebraic theory was motivated by problems arising in scheduling theory, graph theory and operations research (Cuninghame-Green, 1979). Later on, max-plus algebra was also employed in discrete event systems to deal mainly with synchronization problems (Cohen et al., 1985; Baccelli et al., 1992; Cohen et al., 1999; De Schutter and van den Boom, 2008). Other applications include the max-algebraic approach to optimal control (Litvinov et al., 2001; McEneaney, 2006), general max-plus dynamical systems and control (Adzkiya et al., 2015; Hardouin et al., 2011) and generalized HMMs for audiovisual event detection (Maragos and Koutras, 2015; Maragos, 2017). An extensive survey about the applications of the max-plus algebra can be found in Gaubert (2009).

Meanwhile, in the last decade, we have experienced an increase of interest in sparsity in linear equations and linear systems. A sparse solution of a linear equation is the solution with the maximum number of zero elements. The reason we are interested in such solutions, is that they need
less elements to describe the same information. They provide us a way of compressing the available data, throwing away those that are unnecessary (Donoho, 2006). Another important property, is that sparse solutions reveal the structure of partially known signals (Candès et al., 2006) or systems (Chen et al., 2009). In control systems, sparsity has been sought in the sense of minimizing the number of sensors or actuators, subject to energy (Tzoumas et al., 2016; Summers et al., 2016) or observability-controllability constraints (Pequito et al., 2016).

Although sparsity has been extensively studied in the linear setting (Elad, 2010), it is still not much developed in more general nonlinear settings. In this work, we aim to define and study sparsity in the max-plus algebraic setting. A sparse solution of a max-plus equation can be defined as the solution with the maximum number of non-informative elements, i.e. the infinite elements. As in the linear case, such solutions use the least number of elements to describe the same information, thus yielding compressed data. But there are many other applications where sparsity could be relevant. For example, in max-plus systems (either static or dynamical), finding sparse inputs implies that we are activating fewer actuators/machines, thus, saving resources. Similarly, the problem of selecting few sensors to observe a max-plus system could be expressed in terms of designing sparse output matrices. Another application could be in max-plus system identification problems, where the sparsity structure is unknown. In this case, sparse solutions could be employed to reveal the unknown structure of the original system.

To the authors’ best knowledge, this the first work that addresses the following:

i) the definition of sparsity in the max-plus algebraic setting (see Section 3).

ii) the problem of finding the sparsest exact solution of a max-plus equation (see Section 4). We show that the problem is equivalent to the minimum set-cover problem and thus NP-complete.

iii) the problem of finding the sparsest approximate solution (see Section 5). Under some convex constraints, called lateness constraints, we show that the \( \ell_1 \)-error of approximation has supermodular properties. Thus, a greedy approximate algorithm of polynomial complexity can be employed with guaranteed bounds on the ratio of the approximation.

iv) the sparse recovery problem (see Section 6). In particular, it is explored whether we can recover a vector, for which we do not know the sparsity pattern, from its image under a max-plus linear transformation. We derive sufficient conditions, under which, we can use the sparsity framework to recover that vector and its sparsity pattern.

The rest of the paper is organized as follows. In Section 2 we revisit the max-plus equation and its properties. Section 3 formulates the problems of finding the exact and approximate sparsest solutions to the max-plus equation. Then, in Section 4 and Section 5 we present possible solutions to the the former and the latter problem respectively. For completeness, in Section 5 we also include a brief introduction to the supermodularity literature. In Section 6 we study the sparse recovery problem. In Section 7 we study two applications of the sparsity framework to multi-machine interactive production processes (Butkovic, 2010): i) application to resource optimization and ii) application to sparsity-aware system identification. A numerical example of the latter is studied in Section 8. Finally, in Section 9 we conclude the paper and discuss possible extensions of the present work. All proofs which do not appear in the main text are included in the Appendix.
2 Notation and Background

Throughout this paper matrices and vectors will be denoted by bold characters. If $A$ is a $m \times n$ matrix then its columns ($m \times 1$ vectors) are denoted by $A_j$, $j = 1, \ldots, n$. Its components are denoted by $A_{ij}$ or $[A]_{ij}$, for $i = 1, \ldots, m$, $j = 1, \ldots, n$. Similarly, if $x$ is a $n \times 1$ vector, its components are denoted by $x_j$ or $[x]_j$, for $j = 1, \ldots, n$. Finally, for simplicity we denote the row and column index sets by $I = \{1, 2, \ldots, m\}$ and $J = \{1, 2, \ldots, n\}$ respectively.

2.1 The max-plus algebra

The max-plus algebra (or the $(\lor, +)$ semiring) is the set $\mathbb{R}_{\text{max}} = \mathbb{R} \cup \{-\infty\}$ equipped with the maximum operator $\lor$ as “addition” and $+$ as “multiplication” (Gaubert and Plus, 1997). If $x, y \in \mathbb{R}_{\text{max}}$ then $x \lor y \triangleq \max\{x, y\}$. The zero element for the maximum operator $\lor$ is $-\infty$. The operator $+$ is defined in the usual way with $0$ the identity element and $-\infty$ the null element. Similarly, we can define the min-plus algebra on $\mathbb{R}_{\text{min}} = \mathbb{R} \cup \{+\infty\}$, equipped with the minimum operator $\land$ and addition $+$. If $x, y \in \mathbb{R}_{\text{max}}^n$ are vectors, we overload $\lor$ with componentwise maximum $[x \lor y]_i = x_i \lor y_i$, $i = 1, \cdots, n$.

Operators $<, \leq$ are interpreted with the vector partial order, induced by componentwise comparison. We also define the addition $x + a$ of a scalar $a \in \mathbb{R}_{\text{max}}$ to a vector $x \in \mathbb{R}_{\text{max}}^n$ componentwise as follows:

$[x + a]_i = x_i + a$, $i = 1, \cdots, n$.

This can be interpreted as the scalar “multiplication” counterpart of linear algebra.

Next, if $A \in \mathbb{R}_{\text{max}}^{m \times n}$, $B \in \mathbb{R}_{\text{max}}^{m \times n}$ are matrices we define their componentwise maximum $[A \lor B]_{ij} = A_{ij} \lor B_{ij}$, $i = 1, \cdots, m$, $j = 1, \cdots, n$. Finally, we can also define a matrix “multiplication” as in the linear algebra. Let $A \in \mathbb{R}_{\text{max}}^{m \times n}$ and $B \in \mathbb{R}_{\text{max}}^{n \times p}$ then, the max-plus matrix multiplication $A \boxplus B \in \mathbb{R}_{\text{max}}^{m \times p}$ is defined as:

$[A \boxplus B]_{ij} = \bigvee_{k=1}^n (A_{ik} + B_{kj})$, $i = 1, \cdots, m$, $j = 1, \cdots, p$.

2.2 Max-plus linear equation and exact solution

The max-plus equation has a form similar to the linear equation, though we replace addition with maximum and multiplication with addition. In particular, given $A \in \mathbb{R}_{\text{max}}^{m \times n}$, $x \in \mathbb{R}_{\text{max}}^n$, $b \in \mathbb{R}_{\text{max}}^m$, it is given by the following formula:

$\bigvee_{j=1}^n (A_{ij} + x_j) = b_i$, $i = 1, \cdots, m$.

or in compact form

$A \boxplus x = b$. (1)

\footnote{In the general case, $b \in \mathbb{R}_{\text{max}}^m$ (Butkovic, 2010). However, in this paper we will only consider finite $b \in \mathbb{R}^m$. See also Assumption 1 in Section 3}
Next, we define the set of all solutions of (1) \( S(A, b) \):

\[
S(A, b) = \{ x \in \mathbb{R}^n_{\text{max}} : A \boxplus x = b \}
\]  

(2)

We can also write (1) as \( A \boxplus x = \bigvee_{j=1}^n (A_j + x_j) = b \). Hence, in this form, \( A \boxplus x \) can be interpreted as a “max-plus linear combination” of the columns of \( A \) with weights \( x_j \).

To analyze the max-plus equation, we need the definition of the principal solution \( \bar{x} \in \mathbb{R}^n_{\text{min}} \) (Cuninghame-Green, 1979):

\[
\bar{x}_j = \bigwedge_{i=1}^m (b_i - A_{ij}), \forall j \in J.
\]  

(3)

Although the principal solution belongs to \( \mathbb{R}^n_{\text{min}} \), in this paper we will only deal with cases where \( \bar{x} \in \mathbb{R}^n \). When the max-plus equation (1) admits a solution, it turns out that the principal solution \( \bar{x} \) is also an actual solution (see Theorem 7 in the Appendix). In other words, the set \( S(A, b) \) is non-empty if and only if \( \bar{x} \) is a solution to equation (1) (Cuninghame-Green, 1979).

2.3 Max-plus linear equation and approximate solution

Although the principal solution \( \bar{x} \) is always defined, it may not be a solution of (1). In this case, system (1) cannot be solved. However, we may find an approximate solution, by minimizing the \( \ell_1 \) norm of the residual error \( b - A \boxplus x \). Still, without any additional constraint this problem is hard to solve. For this reason, the convex constraint

\[
A \boxplus x \leq b
\]  

(4)

also called the lateness constraint (Cuninghame-Green, 1979), is added to the minimization problem. This relaxation, adopted in Cuninghame-Green (1979), is also motivated by time constraints in operations research (see also Section 7.1). The approximate solution problem can be described with the following optimization problem:

\[
\begin{align*}
\text{minimize} & \quad \| b - A \boxplus x \|_1 \\
\text{subject to} & \quad A \boxplus x \leq b,
\end{align*}
\]  

(5)

which can be recast as a linear program. It turns out that the principal solution \( \bar{x} \) is the largest possible element that satisfies the constraint \( A \boxplus x \leq b \). Therefore, it is also an optimal solution to problem (5) (see Theorem 8 in the Appendix).

3 Problem Statement

In this section, we define the problem of finding the sparsest exact and approximate solutions to equation (1). In linear algebra, the sparsity pattern of a vector or a matrix is determined by the set of its nonzero components. In a similar fashion, in max-plus algebra, the sparsity pattern of any matrix or vector is determined by the set of its finite elements (since the zero element is \(-\infty\)). We define the support of an element \( x \in \mathbb{R}^n_{\text{max}} \) as

\[
\text{supp}(x) = \{ j \in J : x_j \neq -\infty \},
\]
i.e. the set of the indices of its finite components.

The first problem studied in this paper is finding the sparsest solution to equation (1). Formally, given the matrices $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, we want to determine the optimal (possibly non-unique) solution $x^* \in \mathbb{R}^n_{\max}$ to the following optimization problem:

$$x^* = \arg \min_{x \in \mathbb{R}^n_{\max}} |\text{supp}(x)|$$

subject to $A \boxplus x = b$ (6)

where $|\text{supp}(x)|$ denotes the cardinality of the set $\text{supp}(x)$.

However, a solution to (1) may not exist. Meanwhile, solving problem (5) might not work either, since it does not guarantee a sparse approximate solution. Hence, instead of optimizing with respect to the residual error, one option would be to search for sparse approximate solutions to (1), within some allowed error. For this reason, we define an $\epsilon$-approximate solution to (1) as a vector $x \in \mathbb{R}^n_{\max}$ that: i) has residual error bounded by positive constant $\epsilon > 0$ or $\|b - A \boxplus x\|_1 \leq \epsilon$, and ii) satisfies the lateness constraint $A \boxplus x \leq b$.

Thus, in the second problem, given a prescribed constant $\epsilon > 0$, we seek the sparsest (possibly non-unique) $\epsilon$-approximate solution. Equivalently, we solve the optimization problem:

$$x^* = \arg \min_{x \in \mathbb{R}^n_{\max}} |\text{supp}(x)|$$

subject to $\|b - A \boxplus x\|_1 \leq \epsilon$

$$A \boxplus x \leq b$$

(7)

Notice that we may recover the exact sparsest solution problem if we select $\epsilon = 0$.

To guarantee that the problem we are solving is not trivial, we make the following assumption about $A$ and $b$, which holds throughout the paper. It has been a standard assumption in the literature (see chapter 15 in Cuninghame-Green (1979)).

**Assumption 1.** All elements of $b$ in equation (1) are finite: $b \in \mathbb{R}^m$. Every row and column of matrix $A \in \mathbb{R}^{m \times n}_{\max}$ in (1) has at least one finite element$^2$:

- $i$-th row: $\bigvee_{k=1}^n A_{ik} \neq -\infty, \quad i = 1, \ldots, m$
- $j$-th column: $\bigvee_{l=1}^m A_{lj} \neq -\infty, \quad j = 1, \ldots, n$

If this assumption is not satisfied, it leads to trivial situations. For example, if the $k$-th column $A_k$ consists only of $-\infty$ elements, then $x_k$ does not influence the solution at all, since $A_{ik} + x_k = -\infty$, $i = 1, \ldots, m$ for every $x \in \mathbb{R}^n_{\max}$. So, we may remove $k$-th column and variable $x_k$ without any effect.

**Remark 1.** The lateness constraint $A \boxplus x \leq b$ is desirable in many discrete-event systems applications (see also Section [7.1]), where we want some tasks to be completed at times $A \boxplus x$, before some deadlines $b$. In general, it makes problem (7) more tractable. However, in other situations where it is not needed, it might lead to less sparse solutions or higher residual error. It is a subject of future work to explore how we could remove it in problem (7).

$^2$Such matrices are also called doubly $\mathbb{R}$-astic in Butkovic (2010) and doubly G-astic in Cuninghame-Green (1979).
Remark 2. The sparsest solution problem makes sense even if \( m > n \) and we have an overde-termined system. When system (1) is solvable, we might have infinite solutions. Among those solutions some might be sparse.

In the following sections we study the solution to problems (6), (7). Then, we explore the sparse recovery problem as well as applications to multi-machine interactive production processes.

4 Sparsest Exact Solution

In this section we present our results about the solution to the first problem (6). We show that the sparsest solution problem is equivalent to a minimum set cover problem and, thus, NP-complete.

Although the principal solution \( \bar{x} \) defined in (3) is a solution when \( S(A, b) \) is non-empty, it is not sparse, as the next result shows.

Lemma 1. Under Assumption 1, the principal solution \( \bar{x} \) of (1), defined in (3), is finite or \( \bar{x} \in \mathbb{R}^n \).

\[ J_x = \{ j \in J : x_j = \bar{x}_j \} \quad (8) \]

We will call this set the agreement set of \( x \). By Lemma 1 since \( \bar{x} \) if finite, we have \( J_x \subseteq \text{supp}(x) \) for every solution \( x \in S(A, b) \). Thus, the main idea is to try to minimize the agreement set \( |J_x| \). Then, forcing the support to be equal to the agreement set or \( J_x = \text{supp}(x) \) will lead to the sparsest solution.

However, we cannot have arbitrarily small agreement set \( J_x \). There are some necessary conditions that should be satisfied. For each \( j \in J \), we define the row indices \( I_j \subseteq I \), where the minimum in (3) is attained:

\[ I_j = \left\{ i \in I : b_i - A_{ij} = \bigwedge_{k=1}^{m} (b_k - A_{kj}) = \bar{x}_j \right\}. \quad (9) \]

Those necessary conditions require the sub-collection \( I_j, j \in J_x \) to be a set cover of \( I \) (see Theorem 7 in the Appendix).

The next theorem proves that the solution to problem (7) can be reduced to finding the minimum set cover of \( I \), by the subsets \( I_j, j \in J \). On the other hand, any minimum set cover problem can be
reduced to solving an instance of problem (6), for suitably defined matrices $A, b$. Thus, problem (6) is NP-complete.

**Theorem 1.** i) The problem (6) of computing the sparsest max-plus solution is equivalent to finding the minimum set cover of $I$ by the subsets $\{I_j, j \in J\}$ defined in (9). In particular, let $\bar{x}$ be the principal solution defined in (3). Given a minimum set cover $\{I_j, j \in K^* \subseteq J\}$, the element $\hat{x} \in \mathbb{R}^{n}_{\max}$ defined as:

$$
\hat{x}_j = \bar{x}_j, \quad j \in K^*
$$

$$
\hat{x}_j = -\infty, \quad j \in J \setminus K^*,
$$

is an optimal solution to problem (6).

ii) Any minimum set cover problem can be reduced to solving problem (6), for suitably defined matrices $A, b$. Thus, problem (6) is NP-complete.

**Remark 3** (Suboptimal solution to problem (6)). According to Theorem 1, we can solve problem (6), by finding the minimum set cover $\{I_j, j \in K^*\}$ of $I$ and then by using (10) to construct an optimal solution $x^\ast$. Although the minimum set cover is an NP-complete problem, it can be approximated by a greedy algorithm of polynomial complexity with approximation ratio $1 + \log(n)$ (Chvatal, 1979). Alternatively, we could solve problem (6) by solving problem (7) for $\epsilon = 0$, using the techniques of Section 5.

The next example, illustrates the results of this section.

**Example 1.** Suppose we are given the equation

$$
\begin{bmatrix}
1 & 0 & 1 \\
-2 & 2 & 1 \\
1 & 1 & 0
\end{bmatrix} \boxplus \begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} = \begin{bmatrix}
2 \\
0 \\
2
\end{bmatrix}
$$

From (3), the principal solution is

$$
\bar{x} = \begin{bmatrix}
(2 - 1) \land (0 + 2) \land (2 - 1) \\
(2 - 0) \land (0 - 2) \land (2 - 1) \\
(2 - 1) \land (0 - 1) \land (2 - 0)
\end{bmatrix} = \begin{bmatrix}
1 \\
-2 \\
-1
\end{bmatrix}.
$$

From (9), the row index sets $I_j$ are:

$I_1 = \{1, 3\}, I_2 = \{2\}, I_3 = \{2\}$.

The minimum set cover of $I = \{1, 2, 3\}$ is either $I_1 \cup I_2$ or $I_1 \cup I_3$. Hence, we have two possible sparsest solutions: $x_1^\ast = \begin{bmatrix} 1 & -2 & -\infty \end{bmatrix}^T$ and $x_2^\ast = \begin{bmatrix} 1 & -\infty & -1 \end{bmatrix}^T$.

5 Approximate Solution and Supermodular Approach

In this section, we present the approximate solution to problem (7), which uses tools from the supermodular optimization literature; a brief introduction to supermodularity is included in Subsection 5.1. In Subsection 5.2 we reformulate problem (7) to a simpler one, where we only optimize over the support of the optimal solution. Then, in Subsection 5.3 we prove that this new optimization problem has supermodular properties if $A$ has only finite elements (Theorem 3). This
allow us to approximately solve problem (7) via a greedy algorithm of polynomial complexity with guaranteed bounds of approximation (Theorem 4). In some sense, this greedy solution is similar to the “matching pursuit” algorithm in Mallat and Zhang (1993), applied to linear systems. Finally, in Subsection 5.4 we extend the results to the case when matrix $A$ can also have infinite elements (Theorem 5).

5.1 Supermodularity Preliminaries

Supermodularity (Krause and Golovin, 2012) is a property of set functions, that enables us to approximately solve some optimization problems of combinatorial complexity. In particular, greedy algorithms of polynomial complexity can be employed, with theoretical guarantees (bounds) regarding the ratio of approximation (Wolsey, 1982), (Nemhauser et al, 1978). A set function $f : 2^J \rightarrow \mathbb{R}$ is a function that takes a subset $T \subseteq J$ and returns a real value $f(T)$. Two useful properties of set functions are supermodularity and monotonicity. A set function $f : 2^J \rightarrow \mathbb{R}$ is supermodular if for every $C \subseteq B \subseteq J$ and $k \in J$:

$$f(C \cup \{k\}) - f(C) \leq f(B \cup \{k\}) - f(B) \quad (11)$$

Respectively, a set function $f : 2^J \rightarrow \mathbb{R}$ is decreasing if for every $C \subseteq B \subseteq J$, $f(C) \geq f(B)$.

Finally we present a result from Wolsey (1982), which shows how we can approximately solve cardinality minimization problems subject to a supermodular equality constraint. Let the optimization problem be:

$$\begin{array}{ll}
\text{minimize} & |T| \\
\text{subject to} & f(T) = f(J)
\end{array} \quad (12)$$

where $f : 2^J \rightarrow \mathbb{R}$ is supermodular and decreasing, while $|T|$ denotes the cardinality of set $T$. Suppose we use the following greedy algorithm.

Algorithm 1 Greedy Approximate Solution of (12)

1. Set $T_0 = \emptyset$, $k = 0$
2. while $f(T) \neq f(J)$ do
3. \quad $k = k + 1$
4. \quad $j = \arg \min_{s \in J \setminus T_{k-1}} \{ f(T_{k-1} \cup \{s\}) \}$
5. \quad $T_k = T_{k-1} \cup \{j\}$
6. end while
7. return $T_k$

The following theorem provides a bound on the approximation ratio of Algorithm 1.

Theorem 2 (Wolsey (1982)). Suppose $f : 2^J \rightarrow \mathbb{R}$ is supermodular and decreasing. Algorithm 1 returns a suboptimal solution $T_k \subseteq J$ to problem (12) with $|T_k| = k$. If $T^*$ is the optimal solution then the following bound holds

$$\frac{|T_k|}{|T^*|} \leq 1 + \log \left( \frac{f(\emptyset) - f(J)}{f(T_{k-1}) - f(J)} \right) \quad (13)$$

3The result in Wolsey (1982) is for submodular and increasing functions. But $f$ is supermodular (decreasing) if and only if $-f$ is submodular (increasing). Hence, the result is also valid for supermodular and decreasing functions.
In the next sections, we reformulate problem (7) in order to reveal its supermodular structure and leverage the results of Theorem 2.

5.2 Reformulation of Problem (7)

Given any feasible point \( x \) of problem (7), we can construct a new one by forcing every component in the support to be equal to the respective component of the principal solution. In this way, we reduce problem (7) to just finding the support of \( x \), skipping the decision over the finite values of \( x \). Formally, suppose \( x \in \mathbb{R}^n_{\text{max}} \), with support \( \text{supp}(x) = T \), satisfies the inequality \( A \boxplus x \leq b \). Now, define a new element \( z \in \mathbb{R}^n_{\text{max}} \) with the same support as \( x \), \( \text{supp}(z) = T \). Then, replace all its finite components with the ones of the principal solution: \( z_j = \bar{x}_j, j \in \text{supp}(x) \). In terms of the agreement set defined in (8), we have \( J_z = \text{supp}(z) = T \). The next lemma shows that the new vector \( z \) not only is feasible, but also has smaller residual error than \( x \).

**Lemma 2.** Fix a subset \( T \subseteq J \). Let

\[
X_T = \{ x \in \mathbb{R}^n_{\text{max}} : \text{supp}(x) = T, A \boxplus x \leq b \}
\]

be the set of elements which satisfy the lateness constraint and have support equal to \( T \). Assume that \( z \in \mathbb{R}^n_{\text{max}} \) has support and agreement set equal to \( T \):

\[
J_z = T \\
\text{supp}(z) = T.
\]

Then, \( z \in X_T \) and

\[
\| b - A \boxplus x \|_1 \geq \| b - A \boxplus z \|_1,
\]

for all \( x \in X_T \). \( \diamond \)

Since for any fixed support \( \text{supp}(x) = T \subseteq J \), we can select \( x_j = \bar{x}_j, j \in T \) and \( x_j = -\infty, j \in J \setminus T \), the only decision variable that matters in problem (7) is \( T \subseteq J \). To introduce more compact notation, we can rewrite \( A \boxplus x = \bigvee_{j \in J} (A_j + x_j) \) as a max-plus linear combination. But if \( \text{supp}(x) = T \subseteq J \), then this max-plus linear combination becomes:

\[
A \boxplus x = \bigvee_{j \in T} (A_j + x_j), \text{ if } \text{supp}(x) = T.
\]

Choosing \( x_j = -\infty \) is equivalent to ignoring column \( A_j \) in the max-plus linear combination. The next definition uses this notation.

**Definition 1.** We define the **error vector** \( e : 2^J \rightarrow \mathbb{R}^{m}_{\text{min}} \) as:

\[
e(T) = b - \bigvee_{j \in T} (A_j + \bar{x}_j), \text{ for } T \neq \emptyset
\]

\[
e(\emptyset) = \bigvee_{j \in J} e(\{j\}). \tag{14}
\]

\( \diamond \)

4The feasible points of an optimization problem are the elements that satisfy the constraints.
In the definition of \( e(\emptyset) \), operator \( \lor \) is extended to \( \mathbb{R}_{\max} \cup \{+\infty\} \). The \( \ell_1 \)-error function \( E(T) : 2^J \to \mathbb{R}_{\min} \) is defined as the \( \ell_1 \)-norm of the error vector:

\[
E(T) = \| e(T) \|_1. \tag{15}
\]

We note that for the empty set we consider the singletons’ error vectors and take the component-wise maximum in the above definition. This selection guarantees that the \( \ell_1 \)-error function \( E \) is supermodular and decreasing.

The next corollary exploits the result of Lemma 2 and proves that we can rewrite problem (7) as:

\[
\min |T| \text{ subject to } E(T) \leq \epsilon \tag{16}
\]

Corollary 1. Problem (7) is equivalent to problem (16). In particular, if \( \hat{T} \) is an optimal solution to problem (16), then the element \( \hat{x} \in \mathbb{R}^n_{\min} \) defined as:

\[
\hat{x}_j = \bar{x}_j, \quad j \in \hat{T} \\
\hat{x}_j = -\infty, \quad j \in J \setminus \hat{T}, \tag{17}
\]

is an optimal solution to problem (7).

5.3 Finite element case

Now we can show that if \( A \) has only finite elements, the \( \ell_1 \) error set function \( E(T) \), defined in (15), is supermodular.

Theorem 3. Suppose \( A \in \mathbb{R}^{m \times n} \). The \( \ell_1 \) error set function \( E(T) \) defined in (15), is decreasing and supermodular.

The above result along with Corollary 1 enable us to approximately solve problem (7), using Algorithm 2 below. First, we compute the approximate solution to problem (16) in a greedy way. Define \( T_k \subset J \) to be the collection of \( k \) elements, selected greedily in a sequential way. Starting from the empty set \( T_0 = \emptyset \), at each time \( k \), we select the index \( j \), which achieves the smallest \( \ell_1 \)-error \( E(T_{k-1} \cup \{j\}) \). Then, we update \( T_k = T_{k-1} \cup \{j\} \) and this is repeated until the error \( E(T_k) \) becomes less than \( \epsilon \). After the selection of \( T_k \), we construct an approximate solution according to equation (17). The complexity of the algorithm is \( O(n^2) \), since the minimization step requires an inner loop of at most \( n \) iterations, while the outer loop requires at most \( n \) iterations.

Since \( E(T) \) is a supermodular function, it follows that \( \bar{E}(T) = \max (E(T), \epsilon) \) is also supermodular (Krause and Golovin, 2012). Thus, the constraint \( E(T_N) > \epsilon \) is equivalent to \( \bar{E}(T) = \epsilon \). This enables us to leverage the results of Wolsey (1982) (Theorem 2) and provide an upper bound to the ratio of approximation of Algorithm 2.

Theorem 4. Assume that \( A \in \mathbb{R}^{m \times n} \) has only finite elements. Suppose \( \epsilon \geq 0 \) is such that \( E(J) \leq \epsilon \) and \( E(\emptyset) > \epsilon \), where \( E \) is defined in (15). Let \( k \) be the time Algorithm 2 terminates with \( \hat{x}, T_k \) the respective outputs. Then, \( \hat{x} \) is a suboptimal solution to problem (7) with \( T_k = \text{supp}(\hat{x}) \). Moreover, if \( T^* = \text{supp}(x^*) \), where \( x^* \) is the optimal solution of problem (7), the following inequality holds:

\[
\frac{|T_k|}{|T^*|} \leq 1 + \log \left( \frac{m \Delta}{E(T_{k-1}) - \epsilon} \right) \tag{18}
\]
Algorithm 2 Approximate Solution of Problem (7)

Input: $A, b$

1: Compute $\bar{x}$ from (3)
2: if $E(J) > \epsilon$ then
3: return Infeasible
4: end if
5: Initialize $\hat{x}_j = -\infty$, for all $j \in J$
6: Set $T_0 = \emptyset$, $k = 0$
7: while $E(T_k) > \epsilon$ do
8: $k = k + 1$
9: $j = \arg \min_{s \in J \setminus T_{k-1}} E(T_{k-1} \cup \{s\})$
10: $T_k = T_{k-1} \cup \{j\}$
11: end while
12: Update $\hat{x}_j = \bar{x}_j$, $j \in T_k$
13: return $\hat{x}, T_k$

where $\Delta = \bigvee_{i \in I, j \in J} (b_i - A_{ij} - \bar{x}_j)$ and $\bar{x}_j$ are the components of the principal solution defined in (3).

Parameter $\Delta$ is the largest element of the normalized matrix $[b_i - A_{ij} - \bar{x}_j]$, $i \in I, j \in J$. Since $A$ has only finite elements, $\Delta$ is also finite. The presence of the logarithm mitigates the effect of a large $\Delta$ or a small $E(T_{k-1}) - \epsilon$ differences. In general, term $E(T_{k-1})$ depends on $A, b$, but by allowing more memory, it can be precomputed for all possible $k$ with complexity $O(n^2)$ ($O(n)$ per $k$).

Nonetheless, there are special cases, where data independent bounds for the difference $E(T_{k-1}) - \epsilon$ are possible. For example, if both $A$ and $b$ are integer valued, which is common in timing applications, then the error function is also integer valued and $E(T_{k-1}) \geq \lceil \epsilon + 1 \rceil$. Then, the bound of Theorem 4 becomes $\frac{|T_k|}{|T^*|} \leq 1 + \log \left( \frac{m\Delta}{\lceil \epsilon + 1 \rceil - \epsilon} \right)$ and does not depend any more on the specific $A, b$. Quantized elements can also be dealt in a similar fashion.

5.4 Infinite element case

If $A$ has infinite elements, then we cannot directly apply the results of Theorem 2. However, we can replace the infinite elements of the error vector $e(T)$, $T \subseteq J$ with a sufficiently large positive constant $M > 0$ and then exploit the results of the finite case. The idea to replace infinite elements with big constants $M$ is motivated by the “big-M” method in linear optimization (Bertsimas and Tsitsiklis, 1997).

It is sufficient to replace matrix $A \in \mathbb{R}^{m \times n}$ with a new one, denoted by $\hat{A}(M) \in \mathbb{R}^{m \times n}$, where:

$$\hat{A}_{ij}(M) = \begin{cases} A_{ij}, & \text{if } A_{ij} \neq -\infty \\ -M + b_i - \bar{x}_j, & \text{if } A_{ij} = -\infty \end{cases}, \text{ for all } i \in I, j \in J. \quad (19)$$

This new matrix $\hat{A}(M)$ has only finite elements. Thus, we can now apply Algorithm 2 to matrices $\hat{A}(M), b$ instead of $A, b$ and leverage Theorem 4 to bound the approximation ratio. However, we
first have to require that the optimal solution remains the same with this change. This is indeed the case if $M$ is large enough. In particular, if $M > \epsilon$, it turns out that the optimal solution remains the same, as the following lemma shows.

**Lemma 3.** Suppose $M > \epsilon \geq 0$. Then for $\hat{A}(M)$ defined in (19) the following problem:

$$\min_{x \in \mathbb{R}^n} |\text{supp} (x)|$$

subject to $\|b - \hat{A}(M) \oplus x\|_1 \leq \epsilon$

$$\hat{A}(M) \oplus x \leq b$$

is equivalent to problem (7).

Now, we can just apply Theorem 4 to the finite matrices $\hat{A}(M)$ and $b$.

**Theorem 5.** Suppose $M > \epsilon \geq 0$ are constants such that $E(\emptyset) > \epsilon$ and $E(J) \leq \epsilon$, where $E$ is defined in (15). Let $k$ be the time Algorithm 2 terminates under input $\hat{A}(M), b$, where $\hat{A}(M)$ is defined in (19). Let $\hat{x}, T_k$ be the respective outputs. Then, $\hat{x}$ is a suboptimal solution to problem (7) with $T_k = \text{supp}(\hat{x})$. Moreover, if $T^* = \text{supp}(x^*)$, where $x^*$ is the optimal solution of problem (7), the following inequality holds:

$$\frac{|T_k|}{|T^*|} \leq 1 + \log \left( \frac{m \Delta}{\min \{E(T_{k-1}), M\} - \epsilon} \right)$$

where $\Delta = \bigvee_{i \in I, j \in J} (b_i - \hat{A}_{ij}(M) - \bar{x}_j)$.

**Proof.** Let $\hat{E}(T) = b - \bigvee_{j \in T} (\hat{A}(M)_{ij} + \bar{x}_j)$ be the $\ell_1$-error function for $\hat{A}(M), b$. From Theorem 4 and Lemma 3 we obtain:

$$\frac{|T_k|}{|T^*|} \leq 1 + \log \left( \frac{m \Delta}{\hat{E}(T_{k-1}) - \epsilon} \right)$$

But either $\hat{E}(T_{k-1}) = E(T_{k-1})$ if there is no infinite component in $e(T_{k-1})$, or $\hat{E}(T_{k-1}) \geq M$ if there is some infinite component in $e(T_{k-1})$.

**Remark 4.** Notice that:

$$\Delta = \max \{ \bigvee_{i \in I, j \in J, A_{ij} \neq -\infty} (b_i - A_{ij} - \bar{x}_j), M \}.$$

By increasing $M$ we might make the nominator in (21) bigger. Thus, in the sufficient condition $M > \epsilon$ it might be a good choice to select $M$ close to $\epsilon$. On the other hand, we should not choose $M$ too close to $\epsilon$, since we might make the denominator small. In the case of integer valued elements, a reasonable selection could be $M = \epsilon + 1$, since it guarantees $\frac{|T_N|}{|T^*|} \leq 1 + \log \left( \frac{m \Delta}{(\epsilon + 1) - \epsilon} \right)$ as in the finite element case.
6 Application to the Sparse Recovery Problem

In the recovery problem, the goal is to reconstruct some unknown vector \( \mathbf{z} \in \mathbb{R}_{\text{max}}^n \) from the measurements \( \mathbf{A} \oplus \mathbf{z} \in \mathbb{R}^m \), by solving the equation:

\[
\mathbf{A} \oplus \mathbf{x} = \mathbf{A} \oplus \mathbf{z}.
\]

The principal solution \( \bar{\mathbf{x}} \) of the above system could be a good estimate if \( \mathbf{z} \) is dense. However, if the original \( \mathbf{z} \) is sparse then the principal solution misidentifies the \(-\infty\) elements as finite. Here, we explore conditions under which we could use the sparsest solution \( \mathbf{x}^* \) of problem (6) to estimate \( \mathbf{z} \). We call this the sparse recovery problem.

**Problem 1 (Sparse Recovery).** Consider an arbitrary vector \( \mathbf{z} \in \mathbb{R}_{\text{max}}^n \) such that the pair \( (\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A} \oplus \mathbf{z}) \) satisfies Assumption \( \mathcal{A} \). Let \( \mathbf{x}^* \) be the optimal solution of problem (6) for the pair \( (\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A} \oplus \mathbf{z}) \). We say that the Sparse Recovery Problem is solved if \( \mathbf{x}^* \) recovers \( \mathbf{z} \) or \( \mathbf{z} = \mathbf{x}^* \).

This problem is also related to the system identification problem [Schullerus et al., 2006], where, however, the sparsity pattern is considered known.

In the general case, the sparsest solution will have more \(-\infty\) elements than \( \mathbf{z} \). Even if the number of \(-\infty\) elements is correct, the sparsest solution might not be unique as in Example 1. Nonetheless, under some sufficient conditions, the sparse recovery problem can be solved as the next theorem proves.

**Theorem 6.** Consider an element \( \mathbf{z} \in \mathbb{R}_{\text{max}}^n \) such that the pair \( (\mathbf{A}, \mathbf{A} \oplus \mathbf{z}) \) satisfies Assumption \( \mathcal{A} \). Let \( \mathbf{x}^* \) be the optimal solution of problem (6) for \( (\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A} \oplus \mathbf{z}) \). Then, \( \mathbf{x}^* = \mathbf{z} \) if the following sufficient condition holds: For every finite component \( j \in \text{supp}(\mathbf{z}) \), there exists a row index \( i = i(j) \in I \) such that:

a) for all other indices in the support, \( k \in \text{supp}(\mathbf{z}), k \neq j \), we have:

\[
A_{ij} > A_{ik} + z_k - z_j
\]

b) for all indices in the complement of the support, \( l \in J \setminus \text{supp}(\mathbf{z}) \), there exists at least one row index \( s = s(j,l) \in I \), such that:

\[
A_{sl} > A_{il} + [\mathbf{A} \oplus \mathbf{z}]_s - [\mathbf{A} \oplus \mathbf{z}]_i.
\]

Intuitively, the first part of the condition of the preceding theorem states that for any component \( j \in J \) with \( z_j \neq -\infty \), there must be at least one row index \( i \) for which \( A_{ij} \) is large enough, in order to observe the influence of \( z_j \) in \( \mathbf{A} \oplus \mathbf{z} \). Given the previous pair \( (i,j) \), the second part of the condition requires that for every \( l \in J \setminus \text{supp}(\mathbf{z}) \), there exists some row \( s \in I \) such that the component \( A_{sl} \) is large enough to reveal that \( z_l \) is smaller than \( z_j \); small enough to be \(-\infty\).

Both conditions can be guaranteed if, for example, \( m \geq n \) and \( \mathbf{A} \) has large enough leading diagonal elements (or large diagonal elements up to permutations—see Section 8). In this case, if \( A_{jj}, j \in \text{supp}(\mathbf{z}), \) is large enough across the \( j \)-th row then part a) is satisfied with \( i = j \). Similarly if \( A_{ll}, l \in J \setminus \text{supp}(\mathbf{z}), \) is large enough across the \( l \)-th column, then part b) is satisfied by choosing \( s(j,l) = l \) for all \( j \in \text{supp}(\mathbf{z}) \).
7 Applications to discrete event systems

In this section, we give two possible applications of the present framework in discrete-event systems: i) resource optimization; and ii) system identification with unknown sparsity pattern. We motivate those through multi-machine interactive production processes (Butkovic, 2010). Consider \( m \) different products, which are made using \( n \) machines. A machine \( j \in J \) contributes to the completion of a product \( i \in I \) by making a partial product. It processes all partial products in parallel as soon as it starts working. A system matrix \( G \in \mathbb{R}^{m \times n} \) determines how much time it takes for the partial products to be made. Each element \( G_{ij} \) is the time needed for machine \( j \) to make the partial product for product \( i \). Thus, either \( G_{ij} \geq 0 \) or \( G_{ij} = -\infty \) if product \( i \) does not depend on machine \( j \). An input \( u \in \mathbb{R}^n \) describes the times, the machines start working; \( u_j \) is the time, at which machine \( j \) starts working. Similarly, the output \( y = G \square u \) describes the times the products are made; product \( i \) is completed at time \( y_i \). We will use the above model to explore the following problems.

7.1 Resource optimization

Suppose that the products have delivery deadlines \( d \in \mathbb{R}^m \), which should not be exceeded. This implies that the outputs \( y \) should satisfy the lateness constraint \( y \leq d \). Meanwhile, it costs storage resources to make the products before the delivery time. Thus, we wish to restrict the earliness \( \|d - y\|_1 \). Suppose now that we have an extra constraint; we also want to minimize the number of machines used, which consume energy resources. But the number of machines is exactly the support of vector \( u \). Hence, this problem could be formulated as an instance of problem (7) with \( A = G, \ x = u, \ b = d \). Notice that in this case the lateness constraint is not a relaxation but a desired property.

7.2 Sparsity-aware System Identification

Assume we have an unknown system matrix \( G \). Suppose we are given a sequence of \( K \) input-output pairs \((u_l, y_l) \in \mathbb{R}^n_{\max} \times \mathbb{R}^m, \ l \in L = \{1, \ldots, K\}\), which are related via a max-plus product model \( y_l = G \square u_l, \ l \in L \) with \( G \in \mathbb{R}^{m \times n}_{\max} \) (we assume the output is always finite). Our goal is to estimate matrix \( G \) from the given input-output pairs. If we aggregate the latter, we obtain a set of max-plus equations:

\[
\begin{bmatrix}
y_1^\top \\
\vdots \\
y_K^\top \\
y^\top
\end{bmatrix} =
\begin{bmatrix}
u_1^\top \\
\vdots \\
u_K^\top \\
u^\top
\end{bmatrix} \oplus G^\top
\]

or

\[
Y = U \oplus G^\top.
\] (23)

Notice that \( Y \in \mathbb{R}^{K \times m}, \ U \in \mathbb{R}^{K \times n}_{\max} \). System (23) consists of \( m \) separate max-plus equations written together in matrix form.

Now, suppose that we do not know the structure of matrix \( G \). As mentioned before, the principal solution \( \bar{G} \) will have only finite elements, hiding the actual sparsity pattern of the original system.
matrix $G$. Thus, we have to find another way to identify the $-\infty$ elements. One way to approach this problem would be to solve the sparse recovery problem instead. If the sufficient conditions of Theorem $6$ are satisfied, then exact reconstruction is possible. In fact, those conditions also suggest a way to do experiment design, i.e. to design the inputs $U$. It is sufficient to select $U$ with large enough leading diagonal elements such that the sparsest solution recovers $G$. Without knowing $G$, we may not be able to compute how large the leading diagonal elements should be. Nonetheless, we could overcome this problem by exploiting bounds on the finite elements of $G$.

8 Numerical Example

In this section, we present a numerical example, where we apply our results to the system identification problem. Consider a multi-machine interactive production process, as defined in Section $7$, with system matrix $G \in \mathbb{R}^{m \times n}$

$$G = \begin{bmatrix} 2 & 3 & -\infty \\ 1 & 1 & -\infty \\ -\infty & 2 & 6 \end{bmatrix}$$

Now consider $K = 4$ input instances $u_l, l = 1, \ldots, 4$ which are imposed to the system:

$$U^T = \begin{bmatrix} u_1 & \ldots & u_4 \end{bmatrix} = \begin{bmatrix} 0 & 10 & 0 & 2 \\ 10 & 0 & 0 & 0 \\ 5 & 5 & 10 & 2 \end{bmatrix}$$

For each input instance $u_l$, the respective outputs $y_l, l = 1, \ldots, 4$ are:

$$Y^T = \begin{bmatrix} y_1 & \ldots & y_4 \end{bmatrix} = G \boxtimes U^T = \begin{bmatrix} 13 & 12 & 3 & 4 \\ 11 & 11 & 1 & 3 \\ 12 & 11 & 16 & 8 \end{bmatrix}$$

Our goal is to determine $G$ from the given inputs and the corresponding outputs. The principal solution gives:

$$\tilde{G} = \begin{bmatrix} 2 & 3 & -7 \\ 1 & 1 & -9 \\ 1 & 2 & 6 \end{bmatrix}$$

which hides the sparsity pattern of the original matrix $G$. Notice that some $-\infty$ elements in $G$, i.e. $G_{13}$, correspond to negative elements in $\tilde{G}$, i.e. $\tilde{G}_{13} = -7$. Those can be identified as $-\infty$, since $G$, can only have positive or $-\infty$ elements. However, not all of them are negative. For instance, $\tilde{G}_{31} = 1$. Thus, this method does not guarantee that all $-\infty$ elements are revealed.

Suppose now that we compute the sparsest solution $G^*$, by solving problem \cite{6}. In this case we obtain:

$$G^* = G.$$ 

This result is expected, since $U$, is designed to have large diagonal values under the column permutation $\{2, 1, 3\}$ and satisfies the assumptions of Theorem \cite{6}. Thus, without any prior knowledge, we managed to identify for all products, which machines they depend on. If this condition is not satisfied, i.e. if we change $U_{12}$ from 10 to $U_{12} = 1$, then the sparsest solution falsely yields $G^*_{32} = -\infty \neq G_{32}$, but it correctly identifies the remaining elements.
For the next example, suppose that due to some unexpected delay the last output is \( y_4 = [4 \ 2 \ 3 \ 8] \). The equation \( Y = U \ominus G^r \) is no longer satisfied. In this case, we solve problem (7) and find the sparsest approximate solution \( \hat{G} \). For \( \epsilon = 0.3 \), we have \( \hat{G} = G \) and we recover \( G \).

However, if the error gets bigger, for example \( Y_{41} = 5 \), the sparsest approximate solution falsely returns \( G_{72} = -\infty \) for \( \epsilon = 1 \). The results for the sparse recovery problem, presented in Section 6, are only applicable to the exact solution case. Nonetheless, from the last numerical example, it seems that if the delay is small, they might still be valid for the approximate solution case. It is subject of future work to provide a formal analysis.

9 Conclusion

We studied the problem of finding the sparsest solution of the max-plus equation and proved that it is NP-complete. Then we studied the problem of finding the sparsest approximate solution subject to a lateness constraint. The degree of approximation was measured via a \( \ell_1 \) norm function, which was proved to have supermodular properties. Thus, we developed a greedy algorithm of polynomial complexity, which approximates the optimal solution with guaranteed ratio of approximation. We also derived sufficient conditions such that the sparse recovery problem can be solved. The present framework can be applied to discrete event systems applications such as resource optimization or system identification. In future work we will explore whether we can drop the lateness constraint when searching for the sparsest approximate solution. We will also study whether the sufficient conditions of the sparse recovery problem can be relaxed. Another direction is extending the concepts of sparsity to max-plus dynamical systems.

Appendix A: Previous Results

**Theorem 7** (Butković (2003)). An element \( x \in \mathbb{R}^n_{\text{max}} \) is a solution to (1) or \( x \in S(A, b) \) if and only if:

\[
a) \ x \leq \bar{x} \\
b) \bigcup_{j \in J_x} \ I_j = I,
\]

where \( \bar{x} \) is the principal solution defined in (3) and sets \( I_j \) are defined in (9).

**Theorem 8** (Cuninghame-Green (1979)). Let \( \bar{x} \) be the principal solution \( \bar{x} \) defined in (3). The following equivalence holds:

\[
A \oplus x \leq b \Leftrightarrow x \leq \bar{x}.
\]

Moreover, \( \bar{x} \) is an optimal solution to problem (5).

Appendix B: Proofs

**Proof of Theorem 1**

First, we prove \( i \). Suppose that \( x^* \) is an optimal solution to (6). Since it is a solution of the equation \( A \oplus x = b \), by Theorem 7 the subcollection \( \{ I_j, j \in J_{x^*} \} \), determined by the agreement
set \( J_{x^*} = \{ j \in J : x^*_j = \bar{x}_j \} \), is a set cover of \( I \). We will show that the size \(|J_{x^*}|\) of the set cover is minimum. By optimality of \( x^* \), we necessarily have \( x^*_j = -\infty \), for \( j \in J \setminus J_{x^*} \) and the support of \( x^* \) is the same as the agreement set; otherwise, we could create a sparser solution by forcing the elements outside of the agreement set to be \(-\infty\). So, \(|\text{supp}(x^*)| = |J_{x^*}|\). Now, take any set cover \( \{I_j, j \in K \subseteq J\} \) of \( I \) and define element \( x(K) \) as:

\[
\begin{align*}
    x(K)_j &= \bar{x}_j, \quad j \in K \\
    x(K)_j &= -\infty, \quad j \in J \setminus K
\end{align*}
\]

(25)

Notice that \(|\text{supp}(x(K))| = |K|\) and by Theorem 7, \( x(K) \) is also a solution to the max-plus equation \( A \boxplus x = b \). By optimality, \( x^* \) has the smallest support, or \(|\text{supp}(x^*)| \leq |\text{supp}(x(K))|\). But this implies that \(|J_{x^*}| \leq |K|\), which shows that \( \{I_j, j \in J_{x^*}\} \) is a minimum set cover of \( I \).

Conversely, suppose the collection \( \{I_j, j \in K^* \subseteq J\} \) is a minimum set cover of \( I \). Then, we can define the solution \( \hat{x} \) as in (10). We will show that \( \hat{x} \) is an optimal solution to (6). Suppose \( x^* \) is one optimal solution to (6). Then, the collection \( \{I_j, j \in J_{x^*}\} \) is a set cover with \( J_{x^*} = \{ j \in J : x^*_j = \bar{x}_j \} \). Since \( x^* \) is the sparsest solution, we can only have \(|\text{supp}(x^*)| = |J_{x^*}|\). Meanwhile, by optimality of the set cover we have

\[ |\text{supp}(\hat{x})| = |K^*| \leq |J_{x^*}| = |\text{supp}(x^*)| \]

Hence, \( \hat{x} \) is also an optimal solution to (6).

Second, we prove ii). This part is adapted from Butković (2003). Suppose we are given an arbitrary collection of nonempty subsets

\[ S_j \subseteq \{1, \ldots, m\} = I, \quad j \in \{1, \ldots, n\} = J, \]

for some \( m, n \in \mathbb{N} \), such that \( \bigcup_{j \in J} S_j = I \). Define \( A_{ij} = 1 (i \in S_j) \) for all \( i \in I, j \in J \), where \( 1 \) is the indicator function, and \( b_i = 1 \), for all \( i \in I \). By equations (3), (9), it follows that the principal solution is \( \bar{x} = [1 \ldots 1]^T \), while the sets \( S_j \) are equal to the sets \( I_j \). But following the analysis of i), finding the minimum set cover of \( I \) using \( S_j = I_j \) is equivalent to finding the solution to problem (6) with the above selection of \( A, b \). This completes the proof of part ii). \( \square \)

Proof of Lemma 2

By construction, the agreement set and the support are equal to \( T \) or

\[
\begin{align*}
    z_j &= \bar{x}_j, \quad \text{for } j \in T \\
    z_j &= -\infty, \quad \text{for } j \in J \setminus T
\end{align*}
\]

Thus, \( z \leq \bar{x} \) and by Theorem 8 also \( A \boxplus z \leq b \), which proves that \( z \in X_T \).

To prove the second part, again from Theorem 8 if \( x \in X_T \) then

\[
\begin{align*}
    x_j &\leq \bar{x}_j = z_j, \quad \text{for } j \in T \\
    x_j &= z_j = -\infty, \quad \text{for } j \in J \setminus T
\end{align*}
\]

As a result, \( x \leq z \) for any \( x \in X_T \). Now, since \( A \boxplus \cdot \) is increasing (Cuninghame-Green, 1979) we obtain the inequality:

\[ b - A \boxplus z \leq b - A \boxplus x, \]
for any $x \in X_T$. Since both $x, z$ satisfy the lateness constraint (4), we finally have
\[
\|b - A \boxplus x\|_1 = 1^T (b - A \boxplus x) \geq 1^T (b - A \boxplus z) = \|b - A \boxplus z\|_1
\]
for any $x \in X_T$, where $1 = [1 \ldots 1]^T$.

Proof of Corollary 1

Let $x^*, \hat{T}$ be the optimal solutions of problems (7), (16) respectively. Denote by $T^* = \text{supp}(x^*)$ the support of $x^*$. Then construct a new vector $z^*$ such that $z^*_j = \bar{x}_j, j \in T^*$ and $z^*_j = -\infty, i \in J \setminus T^*$. By Lemma 2
\[
E(T^*) = \|b - A \boxplus z^*\|_1 \leq \|b - A \boxplus x^*\|_1 \leq \epsilon.
\]
Thus, $T^* = \text{supp}(x^*)$ is a feasible point of problem (16), implying $|\hat{T}| \leq |T^*|$.

Conversely, define $\hat{x}$ as in (17). By construction and the feasibility of $\hat{T}$ and Lemma 2, we have:
\[
\|b - A \boxplus \hat{x}\|_1 = E(\hat{T}) \leq \epsilon.
\]
\[
A \boxplus \hat{x} \leq b
\]
Thus, $\hat{x}$ is a feasible point of problem (7), which implies $|T^*| \leq |\hat{T}|$. From the above inequalities we obtain $|T^*| = |\hat{T}|$, which also proves that $\hat{x}$ is an optimal solution to problem (7).

Proof of Theorem 3

Notice that we have:
\[
\bigvee_{j \in T} (A_j + \bar{x}_j) \leq \bigvee_{j \in J} (A_j + \bar{x}_j) = A \boxplus \bar{x} \leq b
\]
Thus, we get by construction that the error vector $e(T)$ has only positive components, for every $T \subseteq J$, which implies:
\[
E(T) = \|e(T)\|_1 = 1^T e(T),
\]
where $1^T = [1 \ldots 1]^T$. For convenience, define matrix $\hat{A} \in \mathbb{R}^{m \times n}$ as $\hat{A}_{ij} = A_{ij} + \bar{x}_j$. Then, by the definition (14) of error vector:
\[
\bigvee_{j \in T} \hat{A}_j = b - e(T).
\]

First, we show that $E(T)$ is decreasing. Let $B, C$ be two nonempty subsets of $J$ with $C \subseteq B \subseteq J$. Then, $\bigvee_{j \in C} \hat{A}_j \leq \bigvee_{j \in B} \hat{A}_j$. Consequently, $e(B) \leq e(C)$. Now if $C$ is empty and $B$ is non-empty, then by construction $e(\emptyset) \geq \bigvee_{k \in J} e(\{k\}) \geq e(B)$ (if $C, B$ are both empty we trivially have $e(C) = e(B)$). In any case, by (26), we obtain $E(C) \geq E(B)$.

Second, we show that $E(T)$ is supermodular. Let $C \subseteq B \subseteq J$ and $k \in J \setminus B$. It is sufficient to prove that:
\[
e(C \cup \{k\}) - e(C) \leq e(B \cup \{k\}) - e(B).
\]
For $C \neq \emptyset$ define:

$$ u = \bigvee_{j \in C} \hat{A}_j, \quad v = \bigvee_{j \in C \cup \{k\}} \hat{A}_j $$

$$ z = \bigvee_{j \in B} \hat{A}_j, \quad w = \bigvee_{j \in B \cup \{k\}} \hat{A}_j. $$

By this definition, $v_i = u_i \lor \hat{A}_{ik}$, $w_i = z_i \lor \hat{A}_{ik}$ for every $i \in I$. Also, by monotonicity $u \leq z$, $v \leq w$. There are three possibilities:

i) If $u_i > \hat{A}_{ik}$ then $v_i = u_i$. But also $w_i = z_i$, since by monotonicity $z \geq u$ and $z_i > u_i > \hat{A}_{ik}$. In this case, $v_i - u_i = w_i - z_i = 0$.

ii) If $u_i \leq \hat{A}_{ik}$ and $z_i > \hat{A}_{ik}$ then $v_i = u_i - \hat{A}_{ik} - u_i \geq 0$ and $w_i - z_i = 0 \leq v_i - u_i$.

iii) If both $u_i \leq \hat{A}_{ik}$ and $z_i \leq \hat{A}_{ik}$ then $v_i - u_i = \hat{A}_{ik} - u_i \geq \hat{A}_{ik} - z_i = w_i - z_i$, since by monotonicity $z_i \geq u_i$.

If $C$ is the empty set, we define $u = b - e(\emptyset)$ and $u, z, w$ are defined as before. Since by construction $e(\emptyset) \leq e(k)$ for all $k \in J$, we also have $u \leq v$ and $u \leq w$. Thus, either case ii) or case iii) apply.

In any case, $v - u \geq w - z$ which is equivalent to (27). Finally, multiplying both sides of (27) from the left by $1^T$ gives the desired result: $E(C \cup \{k\}) - E(C) \leq E(B \cup \{k\}) - E(B)$.

**Proof of Theorem 4**

Define the truncated error set function

$$ \bar{E}(T) = \max E(T, \epsilon). $$

By Theorem 3, the error set function $E(T)$ is supermodular and decreasing. Thus, so is the truncated error function (Krause and Golovin 2012). This enables us to express the constraint $E(T) \leq \epsilon$ as $\bar{E}(T) = \bar{E}(J)$. Then, the lines 6–11 of Algorithm 2 are a version of Algorithm 1.

Hence, Theorem 2 readily applies giving the bounds

$$ \frac{|T_k|}{|T|} \leq 1 + \log \frac{\bar{E}(\emptyset) - \bar{E}(J)}{E(T_{k-1}) - E(J)}, $$

where $T$ is the optimal solution of problem (16). From Corollary 1, we can replace $T$ with $T^\ast$. By the assumption $E(\emptyset) > \epsilon$ and the definition of the $\ell_1$-error set function at $\emptyset$:

$$ \bar{E}(\emptyset) = E(\emptyset) = \sum_{i \in I} \bigvee_{j \in J} (b_i - A_{ij} - \bar{x}_j) \leq m \Delta. $$

Meanwhile, we have $E(J) \geq 0$ and the result for the nominator in the logarithm follows. For the denominator, notice that $k$ is such that $E(T_{k-1}) > \epsilon$ and $E(T_k) \leq \epsilon$. Such $k$ exists since $E(J) \leq \epsilon$ and in the worst case, Algorithm 2 halts at $k = |J|$ with $T_k = J$. Thus, we have $E(T_{k-1}) = E(T_{k-1})$ and $\bar{E}(J) = \epsilon$. 

19
Proof of Lemma 3

It is sufficient to prove that the feasible regions of both problems are identical. First, we prove that:

\[ \mathbf{A} \boxplus \mathbf{x} \leq \mathbf{b} \leftrightarrow \hat{\mathbf{A}}(M) \boxplus \mathbf{x} \leq \mathbf{b} \]  

(28)

But from Theorem 8, it is equivalent to show that \( \bar{\mathbf{x}} = \hat{\mathbf{x}} \), where \( \bar{\mathbf{x}} \) is the original principal solution defined in (3) and \( \hat{\mathbf{x}} \) is the new principal solution with \( \hat{\mathbf{A}}(M) \) instead of \( \mathbf{A} \):

\[ \hat{x}_j = \bigwedge_{i=1}^{m} \left( b_i - \hat{A}_{ij}(M) \right), \forall j \in J. \]  

(29)

By construction, \( A_{ij} \leq \hat{A}_{ij}(M) \), which by (3), (29), implies \( \hat{\mathbf{x}} \leq \bar{\mathbf{x}} \). To show the other direction, we have

\[ \hat{x}_j = b_k - \hat{A}_{kj}(M), \text{ for some } k \in I. \]

There are two cases:

i) \( \hat{A}_{kj}(M) = A_{kj} \). Then, \( \hat{x}_j = b_k - A_{kj} \geq \bigwedge_{i \in I} b_i - A_{ij} = \bar{x}_j \).

ii) \( \hat{A}_{kj}(M) = b_k - M - \bar{x}_j \). Then, \( \hat{x}_j = M + \bar{x}_j > \bar{x}_j \), since \( M > 0 \).

Thus, we also have \( \hat{\mathbf{x}} \geq \bar{\mathbf{x}} \). This proves \( \bar{\mathbf{x}} = \hat{\mathbf{x}} \).

Second, we prove that under the constraint \( \mathbf{A} \boxplus \mathbf{x} \leq \mathbf{b} \) (which we showed is equivalent to \( \hat{\mathbf{A}}(M) \boxplus \mathbf{x} \leq \mathbf{b} \)) we have:

\[ \| \mathbf{b} - \mathbf{A} \boxplus \mathbf{x} \|_1 \leq \epsilon \iff \| \mathbf{b} - \hat{\mathbf{A}}(M) \boxplus \mathbf{x} \|_1 \leq \epsilon \]  

(30)

“\( \Rightarrow \)” direction. Since \( A_{ij} \leq \hat{A}_{ij}(M) \), we obtain

\[ \mathbf{A} \boxplus \mathbf{x} \leq \hat{\mathbf{A}}(M) \boxplus \mathbf{x}. \]

But we have \( \mathbf{A} \boxplus \mathbf{x} \leq \mathbf{b}, \hat{\mathbf{A}}(M) \boxplus \mathbf{x} \leq \mathbf{b} \). Thus,

\[ \| \mathbf{b} - \hat{\mathbf{A}}(M) \boxplus \mathbf{x} \|_1 \leq \| \mathbf{b} - \mathbf{A} \boxplus \mathbf{x} \|_1 \leq \epsilon. \]

“\( \Leftarrow \)” direction. For every \( i \in I \), there exists an index \( j_i \in J \) such that:

\[ \| \mathbf{b} - \hat{\mathbf{A}}(M) \boxplus \mathbf{x} \|_1 = \sum_{i=1}^{m} (b_i - \hat{A}_{ij_i}(M) - x_{j_i}) \]

Now assume that some element \( \hat{A}_{kj}(M) \) is equal to \( -M + b_i - \bar{x}_{j_i}, \) for some \( k \in I \). Then, this implies

\[ \epsilon \geq \| \mathbf{b} - \hat{\mathbf{A}}(M) \boxplus \mathbf{x} \|_1 = \sum_{i=1}^{m} (b_i - \hat{A}_{ij_i}(M) - x_{j_i}) \]

\[ \geq b_k - \hat{A}_{kj}(M) - \bar{x}_{j_k} = M, \]
where the second inequality follows from $\hat{A}(M) \boxplus x \leq b$ and the equivalent fact $x \leq \bar{x}$ (see Theorem 8). But since $M > \epsilon$, this is a contradiction and the only possible case is $\hat{A}_{ij_k}(M) = A_{ij}$, for all $i \in I$. Finally,

$$\epsilon \geq \|b - \hat{A}(M) \boxplus x\|_1 = \sum_{i=1}^{m}(b_i - A_{ij_i} - x_{ji})$$

$$\geq \sum_{i=1}^{m}(b_i - \bigvee_{j \in J}(A_{ij} + x_{ji})) = \|b - A \boxplus x\|_1.$$ 

This completes the proof.

Proof of Theorem 6

Define $b = A \boxplus z$. It is sufficient to show that for any solution $x \in \mathbb{R}_{\text{max}}^n$ of equation $A \boxplus x = b$, we have:

$$x_j = z_j, \text{ for all } j \in \text{supp}(z). \quad (31)$$

Then, since $x^*$ is also a solution we have $x^*_j = z_j$, for $j \in \text{supp}(z)$. But $x^*$ is the sparsest solution. Thus, we necessarily have $x^*_j = -\infty$ for $j \notin \text{supp}(z)$; otherwise, $z$ would be a sparser solution contradicting the assumptions. This implies that $z = x^*$.

We now prove (31). Given a $j \in \text{supp}(z)$, let $i = i(j) \in I$ be the row index such that the condition of the theorem holds. Consider the row index sets $I_j \subseteq I$, $j \in J$ defined in (9). Part b) of the condition implies that

$$\bar{x}_l = \bigwedge_{t \in I} b_t - A_{il} \leq b_s - A_{sl} < b_t - A_{il}, \text{ for all } l \in J \setminus \text{supp}(z).$$

This implies that the above minimum is not attained at $i$ or:

$$i \notin I_l, \text{ for all } l \in J \setminus \text{supp}(z). \quad (32)$$

Part a) of the condition implies that

$$b_i = \bigvee_{p \in J}(A_{ip} + z_p) = A_{ij} + z_j.$$ 

Moreover, by the definition (3) of the principal solution:

$$\bar{x}_j = \bigwedge_{q \in I}(b_q - A_{qj}) \leq b_i - A_{ij} = z_j.$$ 

But by Theorem 7 only $\bar{x}_j = z_j$ is possible since the principal solution dominates every other solution. Let $k \in \text{supp}(z)$, $k \neq j$ be another index in the support of $z$. We can similarly show that $\bar{x}_k = z_k$. Now, we claim that $i \notin I_k$. If we had $i \in I_k$, then $z_k = \bar{x}_k = b_i - A_{ik}$ or by replacing $b_i = A_{ij} + z_j$:

$$A_{ij} + z_j = A_{ik} + z_k,$$ 

21
which contradicts the theorem hypothesis $A_{ij} + z_j > A_{ik} + z_k$. Thus:

$$i \not\in I_k, \text{ for all } k \in \text{supp}(z) \setminus \{j\}.$$  \hfill (33)

Since the system $A \boxplus x = b$ is solvable, from (32), (33) $j$ is the unique index such that $i \in I_j$. Hence, set $I$ cannot be covered without including set $I_j$ in the covering. By Theorem[7] any solution $x \in \mathbb{R}^n_{\text{max}}$ must necessarily have $x_j = \bar{x}_j = z_j$.

References

Adzkiya D, De Schutter B, Abate A (2015) Computational techniques for reachability analysis of Max-Plus-Linear systems. Automatica 53:293–302

Baccelli F, Cohen G, Olsder GJ, Quadrat JP (1992) Synchronization and linearity: an algebra for discrete event systems. John Wiley & Sons Ltd

Bertsimas D, Tsitsiklis J (1997) Introduction to linear optimization. Athena Scientific

Butkovič P (2003) Max-Algebra: the linear algebra of combinatorics? Linear Algebra and its Applications 367:313–335

Butkovič P (2010) Max-linear Systems: Theory and Algorithms. Springer

Candès EJ, Romberg J, Tao T (2006) Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on information theory 52(2):489–509

Chen Y, Gu Y, Hero AO (2009) Sparse LMS for system identification. In: 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, pp 3125–3128

Chvatal V (1979) A greedy heuristic for the set-covering problem. Mathematics of operations research 4(3):233–235

Cohen G, Dubois D, Quadrat J, Viot M (1985) A Linear System Theoretic View of Discrete Event Processes and Its Use for Performance Evaluation in Manufacturing. IEEE Trans Automatic Control 30:210–220

Cohen G, Gaubert S, Quadrat J (1999) Max-plus algebra and system theory: Where we are and where to go now. In: Annual Reviews in Control, pp 207–219

Cuninghame-Green R (1979) Minimax Algebra. Springer-Verlag

De Schutter B, van den Boom T (2008) Max-plus algebra and max-plus linear discrete event systems: An introduction. In: Discrete Event Systems, 2008. WODES 2008. 9th International Workshop on, IEEE, pp 36–42

Donoho DL (2006) Compressed sensing. IEEE Transactions on information theory 52(4):1289–1306

Elad M (2010) Sparse and redundant representations. Springer,
Gaubert S (2009) Max-plus Algebraic Tools for Discrete Event Systems, Static Analysis, and Zero-Sum Games. In: International Conference on Formal Modeling and Analysis of Timed Systems, Springer, pp 7–11

Gaubert S, Plus M (1997) Methods and applications of (max,+) linear algebra. In: Reischuk R, Morvan M (eds) STACS 97, Lecture Notes in Computer Science, vol 1200, Springer Berlin Heidelberg, pp 261–282

Hardouin L, Lhommeau M, Shang Y (2011) Towards geometric control of max-plus linear systems with applications to manufacturing systems. In: Proceedings of the IEEE Conference on Decision and Control, pp 1149–1154

Krause A, Golovin D (2012) Submodular function maximization. Tractability: Practical Approaches to Hard Problems 3:19

Litvinov G, Maslov V, Shpiz G (2001) Idempotent functional analysis: An algebraic approach. Mathematical Notes 69(5-6):696–729

Mallat SG, Zhang Z (1993) Matching pursuits with time-frequency dictionaries. IEEE Transactions on signal processing 41(12):3397–3415

Maragos P (2017) Dynamical systems on weighted lattices: general theory. Mathematics of Control, Signals, and Systems

Maragos P, Koutras P (2015) Max-product dynamical systems and applications to audio-visual salient event detection in videos. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, pp 2284–2288

McEneaney WM (2006) Max-plus methods for nonlinear control and estimation. Springer Science & Business Media

Nemhauser GL, Wolsey LA, Fisher ML (1978) An analysis of approximations for maximizing submodular set functions—i. Mathematical Programming 14(1):265–294

Pequito S, Kar S, Aguiar AP (2016) Minimum cost input/output design for large-scale linear structural systems. Automatica 68:384–391

Schullerus G, Krebs V, De Schutter B, van den Boom T (2006) Input signal design for identification of max-plus-linear systems. Automatica 42(6):937 – 943

Summers TH, Cortesi FL, Lygeros J (2016) On submodularity and controllability in complex dynamical networks. IEEE Transactions on Control of Network Systems 3(1):91–101

Tzoumas V, Rahimian MA, Pappas GJ, Jadabaia A (2016) Minimal Actuator Placement With Bounds on Control Effort. IEEE Transactions on Control of Network Systems 3(1):67–78

Wolsey LA (1982) An analysis of the greedy algorithm for the submodular set covering problem. Combinatorica 2(4):385–303