Research Directions: Cyber-Physical Systems

www.cambridge.org/cbp

Result

Cite this article: Berger GO and Sankaranarayanan S (2024). Template-based piecewise affine regression. *Research Directions: Cyber-Physical Systems*. **2**, e5, 1–13. https://doi.org/10.1017/cbp.2024.3

Received: 8 January 2024
Revised: 3 June 2024
Accepted: 21 June 2024

Keywords:
System identification; piecewise affine regression; hybrid systems; algorithms

Corresponding author:
Guillaume O. Berger;
Email: guillaume.berger@uclouvain.be

© The Author(s), 2024. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.

Research Directions

CAMBRIDGE UNIVERSITY PRESS

Template-based piecewise affine regression

Guillaume O. Berger[1] and Sriram Sankaranarayanan[2]

[1]UCLouvain, Louvain-la-Neuve, Belgium and [2]University of Colorado, Boulder, CO, USA

Abstract

We study the problem of fitting a piecewise affine (PWA) function to input–output data. Our algorithm divides the input domain into finitely many regions whose shapes are specified by a user-provided template and such that the input–output data in each region are fit by an affine function within a user-provided error tolerance. We first prove that this problem is NP-hard. Then, we present a top-down algorithmic approach for solving the problem. The algorithm considers subsets of the data points in a systematic manner, trying to fit an affine function for each subset using linear regression. If regression fails on a subset, the algorithm extracts a minimal set of points from the subset (an unsatisfiable core) that is responsible for the failure. The identified core is then used to split the current subset into smaller ones. By combining this top-down scheme with a set-covering algorithm, we derive an overall approach that provides optimal PWA models for a given error tolerance, where optimality refers to minimizing the number of pieces of the PWA model. We demonstrate our approach on three numerical examples that include PWA approximations of a widely used nonlinear insulin–glucose regulation model and a double inverted pendulum with soft contacts.

Introduction

The problem of identifying models from data is central to designing and verifying cyber-physical systems (CPS). These models can predict the output of a subsystem for a given input or the next state of a dynamical system from the current state. Even if there is a physical basis for constructing a model of the system under investigation, it is often necessary to use data-driven modeling to augment these models to incorporate aspects of the system that cannot be easily modeled. CPS are often nonlinear and *multimodal*, wherein different regions of the input/state space have different laws that govern the relationship between the inputs and outputs. In this paper, we study piecewise affine (PWA) regression. The goal of PWA regression is to fit a PWA function to a given data set of input–output pairs $\{(x_k, y_k)\}_{k=1}^{N}$. The PWA function splits the input domain into finitely many regions $H_1, \ldots, H_M$ and associates an affine function $f_i(x) = A_i x + b_i$ to each region H_i. This is illustrated in Figure 1.

Further, we seek a PWA model that fits the given data while respecting a user-provided error bound ε and minimizing the number of regions (pieces). This problem has numerous applications including the identification of hybrid systems with state-based switching and simplifying nonlinear models using PWA approximations. Existing PWA regression approaches usually do not restrict how the input domain is split. For instance, an approach that simply specifies that the input domain is covered by polyhedral sets leads to high computational complexity for the regression algorithm (Lauer and Bloch 2019).

In this paper, we restrict the possible shape of the polyhedral regions by requiring that each region H_i is described by a vector inequality $p(x) \leq c_i$, wherein p is a fixed, user-provided, vector-valued function, called the *template*, while the regions are obtained by varying the offset vector c_i. The resulting problem, called template-based PWA regression, allows us to split the input domain into prespecified shapes using a suitable template. For instance, in Figure 1, the input domain is divided into rectangular regions. Our contributions are as follows:

After introducing and formalizing the problem of template-based PWA regression (Sec. 'Problem Statement'), we show that – similarly to the classical PWA regression problem (Lauer and Bloch 2019) – the problem of template-based PWA regression is NP-hard in the dimension of the input space and the size of the template, but polynomial in the size of the data set (Sec. 'Computational Complexity'). Next, we provide an algorithm for optimal bounded-error template-based PWA regression, i.e., with minimal number of regions while fitting the data within the given error tolerance (Sec. 'Top-Down Algorithm'). Our algorithm is top-down because it breaks large sets of data into smaller ones until those can be fit by an affine function. A more detailed overview of the algorithm is provided later in this introduction, after the 'Related Work'. Finally, we apply our algorithm on two practical problems (Sec. 'Numerical Experiments'): the approximation of a nonlinear system, namely the insulin–glucose regulation process (Dalla Man, Rizza, and Cobelli 2007), with affine functions with rectangular domains, and the identification of a hybrid linear system consisting in an inverted double pendulum with soft contacts on the joints. For both applications, we show that template-based PWA regression

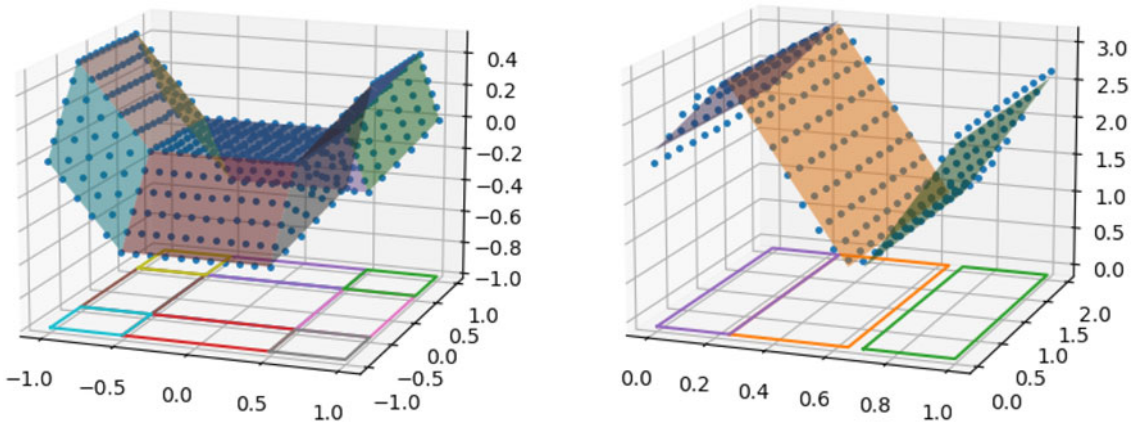


Figure 1. PWA regression of a set of input–output data points with rectangular template.

is favorable compared to classical PWA regression both in terms of computation time and our ability to formulate models from the results. We also compare different templates for the identification of a hybrid linear system consisting of two carts with springs.

This paper is an extension of a preliminary version that appeared as part of the Learning for Decision and Control (L4DC) conference in May 2023 (Berger and Sankaranarayanan 2023). This paper extends our previous version by providing more detailed explanations of the algorithms and complete proofs of all the results. We have also added a new numerical example involving the identification of a hybrid linear system consisting of two carts with springs.

Related work

PWA systems and hybrid linear systems appear naturally in a wide range of applications (Jungers 2009), or as approximations of more complex systems (Breiman 1993). Therefore, the problems of switched affine (SA) and PWA regression have received a lot of attention in the literature; see, e.g., Paoletti et al. (2007) and Lauer and Bloch (2019) for surveys. Both problems are known to be NP-hard (Lauer and Bloch 2019). The problem of SA regression can be formulated as a Mixed-Integer Program (MIP) and solved using MIP solvers, but the complexity is exponential in the number of data points (Paoletti et al. 2007). Vidal et al. (2003) propose an efficient algebraic approach to solve the problem, but it is restricted to noiseless data. Heuristics to solve the problem in the general case include greedy algorithms (Bemporad et al. 2005), continuous relaxations of the MIP (Münz and Krebs 2005), block–coordinate descent (similar to k-mean regression) algorithms (Bradley and Mangasarian 2000; Lauer 2013) and refinement of the algebraic approach using sum-of-squares relaxations (Ozay, Lagoa, and Sznaier 2009); however, these methods offer no guarantees of finding an (optimal) solution to the problem. As for PWA regression, classical approaches include clustering-based methods (Ferrari-Trecate et al. 2003), data classification followed by geometric clustering (Nakada, Takaba, and Katayama 2005) and block–coordinate descent algorithms (Bemporad 2023); however, these methods are not guaranteed to find a (minimal) PWA model. In this regard, our approach considers a novel 'top-down' approach that focuses on searching for subsets of the data that can be part of the same affine model for the given error bounds.

Our approach contrasts with most other approaches in that it is top-down and focuses on refining the domains (using the template assumption) of the pieces until affine fitting is possible. Indeed, most approaches for PWA regression (e.g., Bemporad et al. 2005, Ferrari-Trecate et al. 2003, Bemporad 2023) use a two-step approach in which the data are first clustered by solving a SA regression problem and then the clusters are separated into polyhedral regions. Medhat et al. (2015) and Yuan et al. (2019) use a similar two-step approach for learning hybrid linear automata. There are also approaches that learn the function and the domains in one step: for instance, Breiman (1993) for a special class of continuous PWA functions called *Hinging Hyperplanes*, Sadraddini and Belta (2018) using mixed-integer linear programming (MILP), and Berger, Narasimhamurthy, and Sankaranarayanan (2024) for a class of PWA systems, called guarded linear systems. The class of Hinging Hyperplanes and guarded linear systems are not comparable in general with those studied in this work.[1] Soto, Henzinger, and Schilling (2021) also learn the function and the domains simultaneously for hybrid linear systems; however, their incremental approach is greedy, so that it does not come with guarantees of minimality. By contrast, our approach guarantees to find a PWA function with minimal number of regions from the template.

PWA systems with constraints on the domain appear naturally in several applications including biology (Porreca et al. 2009) and mechanical systems with contact forces (Aydinoglu, Preciado, and Posa 2020), or as approximations of nonlinear systems (Smarra et al. 2020). Techniques for PWA regression with rectangular domains have been proposed in Münz and Krebs (2002) and Smarra et al. (2020); however, these approaches impose further restrictions on the arrangement of the domains of the functions (e.g., forming a grid) and they are not guaranteed to find a solution with a minimal number of pieces. In the one-dimensional case (time series), optimal time series segmentation can be computed efficiently by using dynamic programming (Bellman 1961; Bellman and Roth 1969; Ozay et al. 2012; Ozay 2016), but the approach does not extend to higher dimension and solves a different problem (min. # switches vs. min. # pieces). Our approach bears similarities with the 'split-and-merge' algorithm of Pavlidis and Horowitz (1974), except that (i) we only split regions and never merge them because by construction merging would lead to incompatible regions; (ii) our algorithm comes with guarantees of optimality; and (iii) we address the problem of PWA regression and not segmentation. As for the application involving mechanical

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

systems with contact forces, a recent work by Jin et al. (2022) proposes a heuristic based on minimizing a loss function to learn *linear complementary systems*.

Approach at a glance

Figure 2 below shows the working of our algorithm on a simple data set with $N = 11$ points $(x_k, y_k) \in \mathbb{R} \times \mathbb{R}$ (see Plot I). We seek a PWA function that fits the data with error tolerance $\varepsilon = 0.1$ and with the smallest number of affine pieces (green lines in III, V and VI).

The algorithm works as follows. At the very first step, the approach tries to fit a single straight line through all the 11 points. This corresponds to the *index set* $I_0 = \{1, \ldots, 11\}$ (see II) where the points are indexed as in I. However, no such line can fit the points for the given ε. Hence, our approach generates an *infeasibility certificate* that identifies the indices $C_0 = \{4, 5, 6\}$ as a cause of infeasibility (see II). In other words, we cannot have all three points in C_0 be part of the same piece of the PWA function we seek. As explained in the paper, infeasibility certificates can be computed efficiently using Linear Programming. Moreover, in case of infeasibility, we can always find an infeasibility certificate with cardinality at most $d + 2$, where d is the dimension of the input (here, $d = 1$).[2] Our approach then splits I_0 into two subsets $I_1 = \{1, \ldots, 5\}$ and $I_2 = \{5, \ldots, 11\}$. These two sets are maximal intervals with respect to set inclusion and do not contain C_0. The set I_1 can be fit by a single straight line with tolerance ε (see III). However, considering I_2, we notice once again that a single straight line cannot be fit (see IV). We identify the set $C_2 = \{6, 7, 8\}$ as an infeasibility certificate and our algorithm splits I_2 into maximal subsets $I_3 = \{5, 6, 7\}$ and $I_4 = \{7, \ldots, 11\}$. Each of these subsets can be fit by a straight line (see V and VI). Thus, our approach finishes by discovering three affine pieces that cover all the points $\{1, \ldots, 11\}$. Note that although the data point indexed by 5 is part of two pieces, we can resolve this 'tie' in an arbitrary manner by assigning 5 to the first piece and removing it from the second; the same holds for the data point indexed by 7.

Notation Given two vectors or matrices u and v, their horizontal (resp. vertical) concatenation is denoted by $[u, v]$ (resp. $[u; v]$). For positive integers d and e and a scalar α, we denote by $[\alpha]_d$ (resp. $[\alpha]_{e,d}$) the vector in $\mathbb{R}^d$(resp. matrix in $\mathbb{R}^{e \times d}$) whose components are all equal to α. Finally, given an natural number n, we let $[n] = \{1, \ldots, n\}$.

Problem statement

Given a data set $N \in \mathbb{N}_{>0}$ of input–output pairs $\{(x_k, y_k)\}_{k=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}^e$, the problem of PWA regression aims at finding a PWA function that fits the data within some given error tolerance $\varepsilon \geq 0$. Formally, a PWA function f over a $D \subseteq \mathbb{R}^d$ domain is defined by covering D with M regions $H_1, \ldots, H_M$ and associating an affine function $f_i(x) = A_i x + b_i$ with each H_i:

$$f(x) = \begin{cases} A_1 x + b_1 & \text{if } x \in H_1, \\ A_2 x + b_2 & \text{if } x \in H_2, \\ \vdots & \\ A_M x + b_M & \text{if } x \in H_M. \end{cases}$$

Note that if $H_i \cap H_j \neq \emptyset$ for some $i \neq j$, then f is no longer a function. However, in such a case, we may 'break the tie' by defining $f(x) = f_i(x)$ wherein $i = \min \{j : x \in H_j\}$.

Problem 1. (PWA regression). Given a data set $\{(x_k, y_k)\}_{k=1}^N$ and an error bound $\varepsilon \geq 0$, find M regions $H_i \subseteq \mathbb{R}^d$and affine functions $f_i(x) = A_i x + b_i$ such that

$$\forall\, k,\ \exists\, i : x_k \in H_i \quad \text{and} \quad \forall\, k,\ \forall\, i : x_k \in H_i \Rightarrow \| y_k - f_i(x_k) \|_\infty \leq \varepsilon. \tag{1}$$

Furthermore, we restrict the domain H_i of each affine piece by specifying a *template*, which can be any function $p : \mathbb{R}^d \to \mathbb{R}^h$. Given a template p and a vector $c \in \mathbb{R}^h$, we define the set $H(c)$ as

$$H(c) = \{x \in \mathbb{R}^d : p(x) \leq c\}, \tag{2}$$

wherein $\leq$ is elementwise and $c \in \mathbb{R}^h$ parameterizes the set $H(c)$. We let $\mathcal{H} = \{H(c) : c \in \mathbb{R}^h\}$ denote the set of all regions in $\mathbb{R}^d$described by the template p.

Fixing a template allows to control the complexity of the domains, and thus of the overall PWA function. This allows among others to mitigate overfitting. The *rectangular* template $p(x) = [x; -x]$ defines regions $H(c)$ that form boxes in $\mathbb{R}^d$. Indeed, for two vectors $c_1 \leq c_2$, $H([c_2; -c_1])$ defines the box $\{x \in \mathbb{R}^d : c_1 \leq x \leq c_2\}$. Similarly, allowing pairwise differences between individual variables as components of p yields the *octagon domain* (Miné 2006). Figure 1 illustrates PWA functions with rectangular domains. Thus, we define the template-based piecewise affine (TPWA) regression problem:

Problem 2. (TPWA regression). Given a data set $\{(x_k, y_k)\}_{k=1}^N$, a template $p : \mathbb{R}^d \to \mathbb{R}^h$ and an error bound $\varepsilon > 0$, find M regions $H_i \in \mathcal{H}$ and affine functions $f_i(x) = A_i x + b_i$ *such* that (1) is satisfied.

Prob. 2 can be posed as a *decision problem*: Given a bound $\hat{M}$, is there a TPWA function with $M \leq \hat{M}$ pieces? Alternatively, we can pose it as an optimization problem: Find a TPWA function with minimum number M of pieces.

Although a solution to the decision problem can be used repeatedly to solve the optimization problem, we will focus on directly solving the optimization problem in this paper. Prob. 2 is closely related to the well-known problem of SA regression, in which one aims to explain the data with a finite number of affine functions, but there is no assumption on which function may explain each data point (x_k, y_k). In other words, SA aims to identify a sufficient number of modes and dynamics corresponding to each mode without necessarily explaining how the modes are assigned to each point in the domain.

Problem 3. (SA regression). Given a data set $\{(x_k, y_k)\}_{k=1}^N$ and an error bound $\varepsilon \geq 0$, find M affine functions $f_i(x) = A_i x + b_i$ such that $\forall\, k, \exists\, i$: $\| y_k - f_i(x) \|_\infty \leq \varepsilon$.

Computational complexity

The problem of SA regression (Prob. 3) is known to be NP-hard, even for $M = 2$ (Lauer and Bloch 2019, Sec. 5.2.4). In this section, we show that the same holds for the decision version of Prob. 2. We study the problem in the RAM model, wherein the problem input size is $N(d + e) + h$, where $p : \mathbb{R}^d \to \mathbb{R}^h$.

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

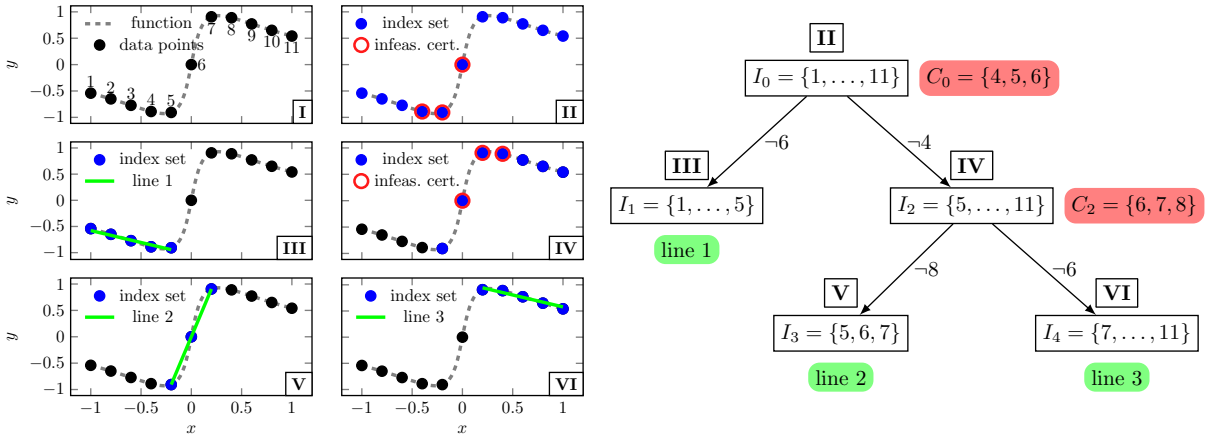


Figure 2. *Left.* Illustration of our algorithm on a simple data set with 11 data points $(x_k, y_k) \in \mathbb{R} \times \mathbb{R}$. *Right.* the index sets explored by our algorithm.

Theorem 1. (NP-hardness). *The decision version of Prob. 2 is NP-hard, even for $M = 2$ and rectangular templates ($p(x) = [x; -x]$).*

The proof reduces Prob. 3 which is known to be NP-hard to Prob. 2.

Proof. Without loss of generality, we restrict to piecewise *linear* models since PWA models can be obtained from linear ones by augmenting each data point x_k with a component equal to 1.

We reduce Prob. 3 to Prob. 2 as follows. Consider an instance of Prob. 3 consisting of a data set $\mathcal{D} = \{(x_k, y_k)\}_{k=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}^e$ and tolerance ε. From $\mathcal{D}$, we build another data set $\mathcal{D}' \subseteq \mathbb{R}^{d+N} \times \mathbb{R}^e$ with $|\mathcal{D}'| = 4N$ as follows. For each $k \in [N]$, we let be the indicator vector of the k^{th} component. We define

$$\mathcal{D}' = \bigcup_{\sigma \in \{-1,1\}} \bigcup_{k=1}^{N} [\{([\sigma x_k; \chi_k], \sigma y_k), ([[0]_d; \chi_k], [\sigma\varepsilon]_e)\}]$$

In other words, for each data point (x_k, y_k) in the original dataset $\mathcal{D}$, we add four data points of the form $([x_k; \chi_k], y_k), ([-x_k; \chi_k], -y_k), ([[0]_d; \chi_k], [\varepsilon]_e), ([[0]_d; \chi_k], [-\varepsilon]_e)$ wherein χ_k is a vector of size N with a 1 entry in the k^{th} position and a 0 entry everywhere else and $[0]_d$ is a vector of d zeros wherein $[\varepsilon]_e$ is a vector with e entries each of which is ε.

Also, we let $\mathcal{P} : \mathbb{R}^{d+N} \to \mathbb{R}^{2(d+N)}$ be the rectangular template in $\mathbb{R}^{d+N}$.

Main step: We show that Prob. 3 with $\mathcal{D}$, ε and $M = 2$ has a solution if and only if Prob. 2 with $\mathcal{D}'$, p, ε and $M = 2$ has a solution.

Proof of 'if direction'. Assume that Prob. 2 has a solution given by and, and for each i, decompose $A_i = [B_i, C_i]$, wherein and . We will show that B_1, B_2 provide a solution to Prob. 3.

Therefore, fix $k \in [N]$. Using the pigeon-hole principle, let $i \in \{1, 2\}$ be such that at least two points in $\{[x_k; \chi_k], [-x_k; \chi_k], [[0]_d; \chi_k]\}$ belong to H_i. Then, by the convexity of H_i, it holds that $[[0]_d; \chi_k] \in H_i$. For definiteness, assume that $[x_k; \chi_k] \in H_i$. Since H_1, H_2 and A_1, A_2 provide a solution to Prob. 2, it follows that

$$\| y_k - B_i x_k - C_i \chi_k \|_\infty \leq \varepsilon,$$
$$\| [\varepsilon]_e - C_i \chi_k \|_\infty \leq \varepsilon, \qquad \| [\varepsilon]_e + C_i \chi_k \|_\infty \leq \varepsilon.$$

The last two conditions imply that $C_i \chi_k = 0$, so that $\| y_k - B_i x_k \|_\infty \leq \varepsilon$. Since k was arbitrary, this shows that B_1, B_2 provide a solution to Prob. 3; thereby proving the 'if direction'.

Proof of 'only if direction'. Assume that Prob. 3 has a solution given by . Then, for each $k \in [N]$, define the intervals as follows: $I_{i,k} = [0, 1]$ if $\| y_k - A_i x_k \|_\infty \leq \varepsilon$, and $I_{i,k} = \{0\}$ otherwise. Now, define the rectangular regions as follows: . Also define the matrices as follows: $B_i = [A_i, [0]_{e,K}]$. We will show that H_1, H_2 and B_1, B_2 provide a solution to Prob. 2.

Therefore, fix $k \in [N]$ and $i \in \{1, 2\}$. First, assume $\| y_k - A_i x_k \|_\infty \leq \varepsilon$. We show that (a) $[x_k; \chi_k]$, $[-x_k; \chi_k]$ and $[[0]_d; \chi_k]$ belong to H_i, and (b)

$$\| y_k - B_i [x_k; \chi_k] \|_\infty \leq \varepsilon, \qquad \| -y_k - B_i [-x_k; \chi_k] \|_\infty \leq \varepsilon,$$
$$\| [\pm\varepsilon]_e - B_i [[0]_d; \chi_k] \|_\infty \leq \varepsilon.$$

This is direct (a) by the definition of $I_{i,k}$, and (b) by the definition of B_i. Now, assume that $\| y_k - A_i x_k \|_\infty \leq \varepsilon$ does not hold. We show that $[x_k; \chi_k]$, $[-x_k; \chi_k]$ do not belong to H_i. This is direct since $1 \notin I_{i,k}$. Thus, we have shown that H_1, H_2 and B_1, B_2 provide a solution to Prob. 2; thereby proving the 'only if direction'.

Hence, we have built a polynomial-time reduction from Prob. 3 to Prob. 2. Since Prob. 3 is NP-hard (Lauer and Bloch 2019, Sec. 5.2.4), this shows that Prob. 2 is NP-hard as well. □

Remark 1. Note that the reduction from Prob. 3 to Prob. 2 in the above proof relies on the fact that $M = 2$. Two comments are due here. First, the fact that Prob. 2 is NP-hard with $M = 2$ implies that Prob. 2 is NP-hard with any $M2$. Indeed, if Prob. 2 can be solved in polynomial time for some $M = \hat{M} > 2$, then one can add spurious data points (e.g., at a far distance of the original data points) to enforce the value of $\hat{M} - 2$ affine pieces of the PWA function. The satisfiability of Prob. 2 with $M = \hat{M}$ and the augmented data set is then equivalent to the satisfiability of Prob. 2 with $M = 2$ and the original data set. Second, given $\hat{M}2$ and any template p, a construction similar to the one used in the above proof can be used to reduce Prob. 3 to Prob. 2 at the cost of introducing a small gap in the reduction. Indeed, fix $\lambda > 0$ and consider the data set . Then, one can show that if Prob. 2 with $\mathcal{D}'$, p, $\varepsilon = \hat{\varepsilon}\left(1 - \frac{2}{\lambda}\right)$ and $M = \hat{M}$ has a solution, then Prob. 3 with $\mathcal{D}$, $\varepsilon = \hat{\varepsilon}$ and $M = \hat{M}$ has a solution. The gap corresponds to the factor $1 - \frac{2}{\lambda}$, which can be made arbitrarily close to one. ◁

So, we showed that Prob. 2 is NP-hard, thereby implying no known algorithm with complexity polynomial in the problem input size $N(d + e) + h$. Nevertheless, one can show that for fixed

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

dimension d, template size h and number of pieces M, the complexity is polynomial in the data size N.

Theorem 2. (Polynomial complexity in N). *For fixed dimension d, template $p : \mathbb{R}^d \to \mathbb{R}^h$ and number of pieces M, the complexity of Prob. 2 is bounded by $O(N^{Mh})$.*

The following notation will be useful in the proof of Theorem 2 below and also later in the paper. For every $c \in \mathbb{R}^h$, let $I(c) = \{k \in [N] : x_k \in H(c)\}$ be the set of all indices k such that $x_k \in H(c)$. Also, let $\mathcal{I} = \{I(c) : c \in \mathbb{R}^h\}$ be the set of all such index sets.

Proof. The crux of the proof is to realize that $|\mathcal{I}| \leq N^h + 1$.

For every $c \in \mathbb{R}^h$, define $P(c) = \{p(x_k) : k \in [N],\ p(x_k) \leq c\}$ and let $\mathcal{P} = \{P(c) : c \in \mathbb{R}^h\}$. First, we prove that $|\mathcal{P}| \leq N^h + 1$. For convenience, we write $p(x) = [p^1(x), \ldots, p^h(x)]$. Note that for any $c \in \mathbb{R}^h$ such that $P(c) \neq \emptyset$, it holds that $P(c) = P(\hat{c})$ where $\hat{c} = \max(\{p(x_k) : k \in [N],\ p(x_k) \leq c\})$ and the 'max' is element-wise. Therefore, we can identify at most h elements $x_{k_1}, \ldots, x_{k_h}$ wherein $k_1, \ldots, k_h \in [N]$ such that $\hat{c} \in \{[p^1(x_{k_1}), \ldots, p^h(x_{k_h})] : k_1, \ldots, k_h \in [N]\}$. Each element can be seen as 'fixing' the maximum value along some dimension of $p(x)$. Hence, there are at most N^h distinct such $\hat{c}$. This implies that there are at most N^h distinct nonempty sets $P(c)$, concluding the proof that $|\mathcal{P}| \leq N^h + 1$.

Next, observe that there is a one-to-one correspondence between $\mathcal{P}$ and $\mathcal{I}$ given by: $P(c) \mapsto I(c)$. Indeed, it is clear that if $I(c_1) = I(c_2)$, then $P(c_1) = P(c_2)$. On the other hand, if $I(c_1) \nsubseteq I(c_2)$, then there is at least one k such that $p(x_k) \leq c_1$ but . This implies that $P(c_1) \nsubseteq P(c_2)$. Therefore, $|\mathcal{P}| = |\mathcal{I}| \in O(N^h)$.

Now, Prob. 2 can be solved by enumerating the $L = N^h$ nonempty index sets $I_1, \ldots, I_L$ in $\mathcal{I}$, and keeping only those I_ℓ for which we can fit an affine function over the data $\{(x_k, y_k)\}_{k \in I_\ell}$ with error bound ε. Next, we enumerate all combinations of M such index sets that cover the indices $[N]$. There are at most L^M such combinations. This concludes the proof of the theorem. □

Remark 2. Note that a similar result holds for Prob. 3 (Lauer and Bloch 2019, Theorem 5.4). The proof of Theorem 2 is however simpler than that in Lauer and Bloch 2019 because in our case we can use the template to build the different regions. ◁

The algorithm presented in the proof of Theorem 2, although polynomial in the size of the dataset, can be quite expensive in practice. For instance, in dimension $d = 2$, with rectangular regions (i.e., $h = 4$) and $N = 100$ data points, one would need to solve $N^h = 10^8$ regression problems,[3] each of which is a linear program.

In the next section, we present an algorithm for TPWA regression that is generally several orders of magnitude faster by using a *top-down* approach that avoids having to systematically enumerate all the elements in $\mathcal{I}$.

Top-down algorithm

We first define the concept of compatible and maximal compatible index sets. We will assume an instance of Prob. 2 with data $\{(x_k, y_k)\}_{k=1}^N$, template p, and error bound ε.

Definition 1. (Maximal compatible index set.) An index set $I \subseteq [N]$ *is compatible* if (a) $I \in \mathcal{I}$ and (b) there is an affine function $f(x) = Ax + b$ such that $\forall\, k \in I,\ \| y_k - f(x_k)\|_\infty \leq \varepsilon$. A compatible index set I is *maximal* if there is no compatible index set I' such that $I \subsetneq I'$.

The key idea is that we can restrict ourselves to searching for *maximal* compatible index sets in order to find a solution to Prob. 2.

Lemma 1. *Let M be given. Prob. 2 has a solution if and only if it has a solution wherein the regions correspond to maximal compatible index sets.*

Proof. The 'if direction' is clear. We prove the 'only if direction'. Consider a solution of Prob. 2 with regions $H_1, \ldots, H_M$. For each $i \in [M]$, there is a maximal compatible index set $I_i = I(c_i)$ such that $H_i \cap \{x_k\}_{k=1}^N \subseteq H(c_i)$. Since $\{x_k\}_{(k=1)}^N \subseteq \cup_{(i=1)}^M H_i$, it holds that $\{x_k\}_{(k=1)}^N \subseteq \cup_{(i=1)}^M H(c_i)$. Hence, $H(c_1), \ldots, H(c_q)$, along with affine functions $f_i(x) = A_i x + b_i$ satisfying (b) in Def. 1, provide a solution to Prob. 2, concluding the proof. □

The main result of this section is that maximal compatible index sets can be computed by using a recursive *top-down* approach as follows (implemented in Algo. 1). Consider the lattice $\langle \mathcal{I}, \subseteq \rangle$ consisting of elements of $\mathcal{I}$ ordered by set inclusion. Our algorithm starts at the very top of follows (implementethis lattice with a set of points $I = [N]$ and descends until we find maximal compatible index sets. At each step, we consider a current set $I \in \mathcal{I}$ (initially, $I = [N]$) that is a candidate for being compatible and check it for compatibility. If I is not compatible, we find subsets $I_1, \ldots, I_S \subsetneq I$ using the FindSubsets procedure, which is required to be *consistent*, as defined below:

Definition 2. (Consistency). Given a noncompatible index set $I \in \mathcal{I}$, a collection of index sets $I_1, \ldots, I_S \in \mathcal{I}$ is said to be *consistent* w.r.t. I if (a) for each s, $I_s \subsetneq I$ and (b) for every compatible index set $J \subseteq I$, there is s such that $J \subseteq I_s$.

We will assume that the procedure FindSubsets is implemented such that for any noncompatible index set I, the collection of sets $I_1, \ldots, I_S$ output by FindSubsets(I) is consistent w.r.t I.

Theorem 3. (Correctness of Algo.1). *If FindSubsets satisfies that for every noncompatible index set $I \in \mathcal{I}$, the output of* FindSubsets*(I) is consistent w.r.t. I, then Algo. 1 always terminates and the output $\mathcal{S}$ contains all the maximal compatible index sets.*

Proof. Termination follows from the fact that each index set $I \in \mathcal{I}$ is picked at most once, because when some $I \in \mathcal{I}$ is picked, it is then added to the collection $\mathcal{V}$ of visited index sets, so that it cannot be added a second time to $\mathcal{U}$ (line 12). Since $\mathcal{I}$ is finite, this implies that the algorithm terminates in a finite number of steps.

Now, we prove that, upon termination, any maximal compatible index set is in the output $\mathcal{S}$ of the algorithm. Therefore, let J be a maximal compatible index set. Then, among all sets I picked during the execution of the algorithm and satisfying $J \subseteq I$, let I^* have minimal cardinality. Such an index set exists since $J \subseteq [N]$. We will show that:

Main result. $I^* = J$.

Proof of main result. For a proof by contradiction, assume that $I^* \neq J$. Since J is maximal and $J \subsetneq I^*$, I^* is not compatible. Hence, the index sets $(I_1, \ldots, I_S) = \mathit{FindSubsets}(I^*)$ were added to $\mathcal{U}$ (line 11). Using the assumption on FindSubsets, let s be such that $J \subseteq I_s \subsetneq I^*$. Since I_s must have been picked during the execution of the algorithm, this contradicts the minimality of the cardinality of I^*, concluding the proof of the main result.

Thus, J was picked during the execution of the algorithm. Since it is compatible, it was added to $\mathcal{S}$ at the iteration at which it was picked (line 8), and since it is maximal, it is not removed at later iterations. Hence, upon termination, $J \in \mathcal{S}$. Since J was arbitrary,

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

Algorithm 1. Top-down algorithm to compute maximal compatible index sets

Data: Data set $\{(x_k, y_k)\}_{k=1}^N$, template p
Result: Collection $\mathcal{S}$ of all maximal compatible index sets
1 $\mathcal{S} \leftarrow \emptyset$ // compatible sets so far
2 $\mathcal{U} \leftarrow \{[N]\}$ // sets to be processed
3 $\mathcal{V} \leftarrow \emptyset$ // already explored
4 **while** $\mathcal{U} \neq \emptyset$ **do**
5 Pop an index set I from $\mathcal{U}$ // removes I from $\mathcal{U}$
6 **if** *I is a subset of a set of $\mathcal{S}$* **then**
// do nothing
7 **else if** *I is compatible* **then**
8 Add I to $\mathcal{S}$
9 Remove strict subsets of I from $\mathcal{S}$ // Removed sets are not maximal
10 **else**
11 $(I_1, \ldots, I_S) \leftarrow$ FindSubsets(I) // must be consistent
12 Add to $\mathcal{U}$ all $I_1, \ldots, I_S$ that are not in $\mathcal{V}$
13 **end**
14 Add I to $\mathcal{V}$
15 **end**
16 **return** $\mathcal{S}$

Algorithm 2. An implementation of FindSubsets using infeasibility certificates

Data: Data set $\{(x_k, y_k)\}_{k=1}^N$, template $p = [p^1, \ldots, p^h]$, non-compatible index set $I = I(c)$ where $c = [c^1, \ldots, c^h]$, infeasibility certificate $C \subseteq I$
Result: A collection of index sets $I_1, \ldots, I_S$ consistent w.r.t. I
1 **foreach** $s = 1, \ldots, h$ **do**
2 $\hat{c}^s \leftarrow \max\{p^s(x_k) : k \in I,\ p^s(x_k) < \max_{x \in C} p^s(x)\}$
3 Define $I_s = I([c^1, \ldots, c^{s-1}, \hat{c}^s, c^{s+1}, \ldots, c^h])$
4 **end**
5 **return** all nonempty index sets $I_1, \ldots, I_h$

we conclude that upon termination, $\mathcal{S}$ contains all maximal compatible index sets.

Finally, we show that upon termination, $\mathcal{S}$ contains only maximal compatible index sets. This follows from the fact that, at each iteration of the algorithm, for any distinct $I_1, I_2 \in \mathcal{S}$, it holds that $I_1 \nsubseteq I_2$ and $I_2 \nsubseteq I_1$. Indeed, when I_1 is added to $\mathcal{S}$, all subsets of I_1 are removed from $\mathcal{S}$ (line 9) and are ignored at later iterations (line 6). The same holds for I_2. This concludes the proof of the theorem. □

Implementation of FindSubsets using infeasibility certificates

We now explain how to implement FindSubsets so that it is consistent. For that, we use infeasibility certificates, which are index sets that are not compatible.

Definition 3. (Infeasibility certificate). An index set $C \subseteq [N]$ is an infeasibility certificate if there is no affine function $f(x) = Ax + b$ such that $\forall\, k \in C, \| y_k - f(x_k)\|_\infty \leq \varepsilon$.

Note that any incompatible index set I contains an infeasibility certificate $C \subseteq I$ (e.g., $C = I$). However, it is quite useful to extract an infeasibility certificate C that is as small as possible. We explain below how to compute small infeasibility certificates. Thereafter, from an infeasibility certificate $C \subseteq I$, one can compute a consistent collection of index subsets of I by tightening each component of the template *independently*, in order to exclude a minimal nonzero number of indices from the infeasibility certificate, while keeping the other components unchanged. This results in an implementation of FindSubsets that satisfies the consistency property, described in Algo. 2. Figure 3 below shows an illustration for rectangular regions.

Theorem 4. (Correctness of Algo. 2). *For every noncompatible index set $I \in \mathcal{I}$, the output $I_1, \ldots, I_S$ of Algo. 2 is consistent w.r.t. I.*

Proof. Observe that if C is an infeasibility certificate, then every $I \subseteq [N]$ satisfying $C \subseteq I$ is not compatible. Now, let $J \subseteq I$ be compatible. Using that $C \nsubseteq J$, let s be a component such that $\max_{k \in J} p^s(x_k) < \max_{k \in C} p^s(x_k)$. It holds that $J \subseteq I_s$. Since J was arbitrary, this concludes the proof. □

Finding infeasibility certificates

We outline the process of finding an infeasibility certificate $C \subseteq I$ for a given noncompatible index set $I \subseteq [N]$. Recall that the data is

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

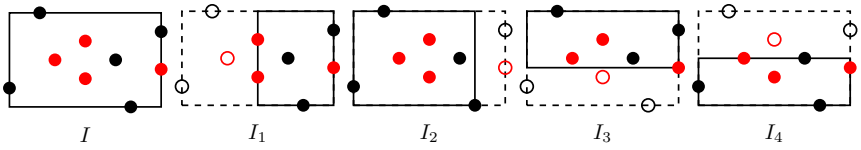


Figure 3. FindSubsets implemented by Algo. 2 with rectangular regions. The red dots represent the infeasibility certificate C. Each I_s excludes at least one point from C by moving one face of the box but keeping the others unchanged.

of the form $\{(x_k, y_k)\}_{k=1}^N$, wherein for each $k \in [N]$, $x_k \in \mathbb{R}^d$ and $y_k \in \mathbb{R}^e$. For simplicity, assume that the output is scalar, i.e., $e = 1$, or equivalently, $y_k \in \mathbb{R}$ for all $k \in [N]$. We will subsequently show how technique for the scalar case will extend to the case of $e > 1$.

For the scalar case, the goal is to find an affine function $f(x) = a^{\mathrm{T}}x + b$ wherein $a \in \mathbb{R}^d$ and $b \in \mathbb{R}$ so that for all $k \in I$, $|y_k - f(x_k)| \leq \varepsilon$. However, since I is noncompatible, no such function exists by definition. Therefore, the system of linear inequalities involving unknowns $(a, b) \in \mathbb{R}^d \times \mathbb{R}$ is infeasible:

$$\begin{array}{rcll} x_k^{\mathrm{T}}a + b - y_k & \leq & \varepsilon & \forall\, k \in I, \\ -x_k^{\mathrm{T}}a - b + y_k & \leq & \varepsilon & \forall\, k \in I. \end{array}$$

Note that each constraint of the form $|\alpha| \leq \beta$ is expanded as two constraints $\alpha \leq \beta$ and $-\alpha \leq \beta$. By applying Farkas' Lemma or a theorem of the alternative (Rockafellar 1970, Theorem 21.3), and simplifying the result, we conclude that the system of inequalities above is infeasible (i.e., I is noncompatible) if and only if there exists a multiplier $\lambda_k \in \mathbb{R}$ for each $k \in I$ such that the following system of linear constraints is feasible:

$$\begin{array}{l} \sum_{k\in I} \lambda_k x_k = [0]_d, \\ \sum_{k\in I} \lambda_k = 0, \\ \sum_{k\in I} \lambda_k y_k + \sum_{k\in I} |\lambda_k| \varepsilon \leq -1\ . \end{array} \tag{3}$$

Thus, in order to check whether a given index set I is noncompatible, we simply formulate the system of inequalities (3) involving unknowns $\lambda_k \in \mathbb{R}$ for each $k \in I$ and attempt to find a feasible solution using a Linear Programming (LP) solver. If feasible, we conclude that I is noncompatible. In fact, given any solution $(\lambda_k)_{k\in I}$ to the system of inequalities above, we can extract a corresponding infeasibility certificate as $C = \{k \in I : \lambda_k \neq 0\}$. It is easy to see why: any $k \in I$ with $\lambda_k = 0$ indicates that $I \backslash \{k\}$ remains a noncompatible set since such a variable λ_k can be removed from the system of inequalities while retaining feasibility.

Lemma 2. *If I is noncompatible, then there exists an infeasibility certificate $C \subseteq I$ such that $|C| \leq d + 2$.*

Proof. The system of constraints (3) has $|I|$ unknowns and $d + 2$ constraints of which $d + 1$ are equality constraints. We may write $\lambda_k = \alpha_k - \beta_k$ for variables $\alpha_k, \beta_k \geq 0$ and therefore $|\lambda_k| = \alpha_k + \beta_k$. Under this transformation, the system of constraints can be rewritten as

$$\begin{array}{l} \sum_{k\in I} (\alpha_k - \beta_k) x_k = [0]_d, \\ \sum_{k\in I} (\alpha_k - \beta_k) = 0, \\ \sum_{k\in I} (\alpha_k - \beta_k)\gamma_k + \sum_{k\in I} (\alpha_k + \beta_k)\varepsilon \leq -1, \\ (\alpha_k, \beta_k) \geq 0, \forall k \in I \end{array}$$

With this transformation, the system above is a standard-form Linear Program with $d + 2$ constraints and decision variables $\alpha_k, \beta_k 0$ for each $k \in I$. We treat the objective function for this LP as the constant 0. Any basic feasible solution will have at most $d + 2$ nonzero variables. Furthermore, from the theory of linear programming, we know that if a system is feasible, then it has a basic feasible solution (Vanderbei 2020). Translating this back to the original system (3), we get that there is a solution involving at most $d + 2$ nonzero values for λ_k. That is, there exists an infeasibility certificate C such that $|C| \leq d + 2$. □

So far, we have assumed that the outputs y_k are scalar, i.e., $e = 1$. However, if $e > 1$, we can simply use the previous analysis by focusing separately on each component of the output vectors y_k. This is possible because if a given index set I is noncompatible, then there must exist a component $s \in [e]$ such that I is noncompatible even if the data set is restricted to $\{(x_k, y_k^s)\}_{k=1}^N$, wherein y_k^s denotes the s^{th} component of y_k. Indeed, if it were not the case, then for each component s, we could find an affine function $f_s(x) = a_s^{\mathrm{T}}x + b_s$ such that for all $k \in I$, $|f_s(x_k) - y_k^s| \leq \varepsilon$. By letting $A = [a_1^{\mathrm{T}} \ldots; a_e^{\mathrm{T}}]$ and $b = [b_1, \ldots, b_e]^{\mathrm{T}}$, we see that $f(x) = Ax + b$ satisfies that for all $k \in I$, $\| f(x_k) - y_k \|_\infty \leq \varepsilon$. This contradicts the assumption that I is noncompatible. Thus, the infeasibility certificate for $e > 1$ is extracted by simply considering each output component in turn, thus reducing the problem to the scalar case considered above.

We conclude that checking whether a set I is noncompatible and if so, finding an infeasibility certificate $C \subseteq I$, can be solved by posing the system of constraints (3) and solving it using an algorithm such as the simplex algorithm.

Good infeasibility certificates for the top-down approach

The implementation of FindSubsets boils down to finding infeasibility certificates, which can be done as explained above. However, not all certificates will be as good in terms of overall complexity of the top-down approach. To exclude noncompatible index sets more rapidly, it is desirable that the points in the certificate are 'spatially concentrated' in the input domain. This means that the points $\{x_k\}_{k\in C}$ are close to each other w.r.t. some distance metric.

We illustrate the benefit of spatially concentrated certificates with the example used to illustrate the top-down approach in the introduction.

Example 1. Consider the TPWA regression problem in Figure 2, introduced in the section 'Approach at a glance'. In Plot II, we obtained the infeasibility certificate $C_0 = \{4, 5, 6\}$. Note that $\hat{C}_0 = \{1, 5, 11\}$ is another infeasibilitty certificate that could have been obtained as a result of solving the system (3). However, C_0 is more 'spatially concentrated' in the sense that the points in C_0 are closer to each other than those in $\hat{C}_0$.

Recall that using C_0 as the infeasibility certificate allowed to find the compatible subset $I_1 = \{1, \ldots, 5\}$ directly and the compatible subsets $I_3 = \{5, 6, 7\}$ and $I_4 = \{7, \ldots, 11\}$ at the subsequent steps.

However, if the certificate $\hat{C}_0 = \{1, 5, 11\}$ were used, then we would have obtained $\hat{I}_1 = \{2, \ldots, 11\}$ and $\hat{I}_2 = \{1, \ldots, 10\}$. Note that both $\hat{I}_1, \hat{I}_2$ are noncompatible because they contain C_0 and thus further steps of our procedure are needed until we find maximally compatible sets. ◁

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

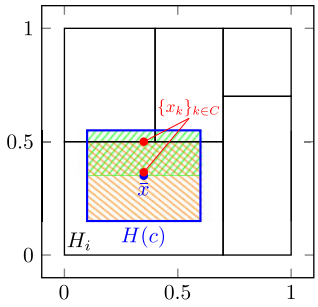


Figure 4. Illustration of FindSubsets with a spatially concentrated certificate. The green and orange hatched rectangles illustrate two possible cases for $H(c_s)$ output by FindSubsets.

We now refine the above argument with a volume-contraction argument to discuss what would be the complexity of the overall top-down algorithm if all certificates are spatially concentrated in the input domain.

Example 2. Consider a PWA function f with M pieces whose domains $H_1, \dots, H_M$ are rectangles, as illustrated in Figure 4. Let $N \in \mathbb{N}_{>0}$ and consider a data set $\mathcal{D} = \{(x_k, y_k)\}_{k=1}^N$. We aim to solve Prob. 2 with $\mathcal{D}$, $\varepsilon = 0$ and the rectangular template. We will discuss the complexity of the top-down approach presented in Algo. 1 if all certificates are spatially concentrated. This means that $\{x_k\}_{k\in C}$ consists in $d+2$ points concentrated around the center $\bar{x}$ of $H(c)$, where C is the certificate for $I(c)$ (see Figure 4).

This will imply that the rectangles $H(c_1), \dots, H(c_h)$ computed by FindSubsets satisfy that for all $s \in [h]$, either the volume of $H(c_s)$ is half of that of $H(c)$ since one face is tight at $\bar{x}$ (see the green rectangle in Figure 4) or $H(c_s)$ has one more face near the boundary of H_i compared to $H(c)$ (see the orange rectangle in Figure 4). By adding the natural assumption that all regions H_j have a volume of at least $\nu \in (0, 1]$ and discarding regions with volume smaller than ν, we get that the algorithm cannot divide the volume of a region more than $-\log_2(\nu)$ times. Hence, the depth of the tree underlying the algorithm is upper bounded by $h - \log_2(\nu)$. Since, each node of the tree has at most h children (the subsets given by FindSubsets), the number of rectangles encountered during the algorithm is upper bounded by $h^{h-\log_2(\nu)}$. Note that this upper bound on the complexity of the algorithm is independent of the data size N. This concludes the example. ◁

To conclude this section on good certificates, we explain briefly how spatially concentrated certificates can be computed by adding a cost function to the Linear Program (3). For simplicity, we will assume that the output dimension $e = 1$. For the case when $e > 1$, we can apply our approach to each component of the output in turn.

Given a center point $\bar{x} = \frac{1}{N}\sum_{i\in I} x_i$ for a noncompatible index set $I \subseteq [N]$, we consider the following Linear Program with variables $\lambda_k \in \mathbb{R}$ for each $k \in I$,

$$\begin{array}{ll} \text{minimize} & \sum_{k\in I} |\lambda_k| \parallel x_k - \bar{x} \parallel^2 \\ \text{s.t.} & \sum_{k\in I} \lambda_k x_k = [0]_d, \\ & \sum_{k\in I} \lambda_k = 0, \\ & \sum_{k\in I} \lambda_k y_k + \sum_{k\in I} |\lambda_k| \varepsilon \le -1 \ . \end{array} \tag{4}$$

The objective function of (4) tends to put zero value to λ_k whenever $\parallel x_k - \bar{x} \parallel$ is large. This promotes proximity of the point x_k to $\bar{x}$ when $\lambda_k \neq 0$.[4]

Early stopping using set cover algorithms

Finally, Algo. 1 can be made more efficient by enabling early termination if $[N]$ is optimally covered by the compatible index sets computed so far. For that, we add an extra step at the beginning of each iteration, that consists in (i) computing a lower bound β on the size of an optimal cover of $[N]$ with compatible index sets and (ii) checking whether we can extract from $\mathcal{S}$ a collection of β index sets that form a cover of $[N]$. The extra step returns *break* if (ii) is successful. An implementation of the extra step is provided in Algo. 3.

The soundness of Algo. 3 follows from the following lemma.

Lemma 3. *Let β be as in Algo. 3. Then, any cover of $[N]$ with compatible index sets has size at least β.*

Proof. The crux of the proof relies on the observation from the proof of Theorem 3 that for any compatible index set $I \in \mathcal{I}$, there is $J \in \mathcal{S} \cup \mathcal{U}$ such that $I \subseteq J$. It follows that for any cover of $[N]$ with M compatible index sets, there is a cover of $[N]$ with M index sets in $\mathcal{S} \cup \mathcal{U}$. Since β is the smallest size of such a cover, this concludes the proof. □

The implementation of the extra step in Algo. 1 is provided in Algo. 4. The correctness of the algorithm follows from that of Algo. 1 (Theorem 3) and Algo. 3 (Lemma 3). In conclusion, we provided an algorithm for optimal TPWA regression.

Theorem 5. (Optimal TPWA regression). *Algo. 4 solves Prob. 2 with minimal M.*

Proof. Let $I_1, \dots, I_M$ be the output of Algo. 4. For each $i \in [M]$, let $H_i = H(c_i)$ where $I_i = I(c_i)$ and let $f_i(x) = A_i x + b_i$ be as in (b) of Def. 1. The fact that $H_1, \dots, H_M$ and $f_1, \dots, f_M$ is a solution to Prob. 2 follows from the fact that $I_1, \dots, I_M$ is a cover of $[N]$ and the definition of $f_1, \dots, f_M$. The fact that it is a solution with minimal M follows from the optimality of $I_1, \dots, I_M$ among all covers of $[N]$ with compatible index sets. □

Remark 3. To solve the optimal set cover problems (known to be NP-hard in general) in Algo. 3, we use MILP formulations. The complexity of solving these MILPs grows as $2^{|\mathcal{S}|}$ and $2^{|\mathcal{S}\cup\mathcal{U}|}$, respectively. However, in our numerical experiments (see next section), we observed that the gain of stopping the algorithm early if an optimal cover is found systematically outbalanced the computational cost of solving the set cover problems. Furthermore, if one is satisfied with a suboptimal solution, they can use an approximation algorithm, such as the greedy algorithm, which outputs a cover whose size is within some factor $t(N) \ge 1$ of the optimal set cover size (Chvatal 1979). In this case, Algo. 3 outputs *break* if $\alpha \le t(N)\beta$. ◁

Numerical experiments

In this section, we demonstrate the applicability of our algorithm on three numerical examples.[5] We also compare it with the MILP and Piecewise Affine Regression and Classification (PARC) (Bemporad 2023) approaches to solve SA and PWA regression, and we discuss the impact of different templates in terms of simplicity of the model and efficiency of the algorithm.

PWA approximation of insulin–glucose regulation model

Dalla Man, Rizza, and Cobelli (2007) present a nonlinear model of insulin–glucose regulation that has been widely used to test artificial pancreas devices for treatment of type-1 diabetes. The model is nonlinear and involves 10 state variables. However, the

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

Algorithm 3. Extra step at the beginning of each iteration of Algo. 1

Data: $\mathcal{S}$ and $\mathcal{U}$ at the iteration, N
Result: BREAK if we can extract from $\mathcal{S}$ an optimal cover of $[N]$ with compatible index sets; otherwise, CONTINUE
Let α be the size of an optimal cover of $[N]$ by index sets in $\mathcal{S}$ `// Note: α = ∞ if [N] cannot be covered by S.`
Let β be the size of an optimal cover of $[N]$ by index sets in $\mathcal{S} \cup \mathcal{U}$
if $\alpha \leq \beta$ **then return** BREAK **else return** CONTINUE

Algorithm 4. Top-down algorithm for Prob. 2.

```
// Lines 1-3 from Algo. 1
while true do
    if Algo. 3 outputs BREAK then return an optimal cover of [N] using index sets from S
    // Lines 5-14 from Algo. 1
end
```

nonlinearity arises mainly from the term U_{id} (insulin-dependent glucose utilization) involving two state variables, say x_1 and x_2 (namely, the level of insulin in the interstitial fluid, and the glucose mass in rapidly equilibrating tissue):

$$U_{\text{id}}(x_1, x_2) = \frac{(3.2667 + 0.0313x_1)x_2}{253.52 + x_2}.$$

We consider the problem of approximating U_{id} with a PWA model, thus converting the entire model into a PWA model. Therefore, we simulated trajectories and collected $N = 100$ values of x_1, x_2 and $U_{\text{id}}(x_1, x_2)$; see Figure 5(a). For three different values of the error tolerance, $\varepsilon \in \{0.2, 0.1, 0.05\}$, we used Algo. 4 to compute a PWA regression of the data with rectangular domains. The results of the computations are shown in Figure 5(b,c,d). The computation times are, respectively, 1, 22, and 112 s. Finally, we evaluate the accuracy of the PWA regression for the modeling of the glucose-insulin evolution by simulating the system with U_{id} replaced by the PWA models. The results are shown in Figure 5(e, f). We see that the PWA model with $\varepsilon = 0.05$ induces a prediction error less than 2 % over the whole simulation interval, which is a significant improvement compared to the PWA models with only 1 affine piece ($\varepsilon = 0.2$) or 2 affine pieces ($\varepsilon = 0.1$).

Finally, we compare with SA regression and classical PWA regression. To find a SA model, we solved Prob. 3 with $\varepsilon = 0.05$ and $M = 3$ using a MILP approach. The computation is very fast (< 0.5 s); however, the computed clusters of data points (see Figure 6) do not allow to learn a (simple) PWA model, thereby hindering the derivation of a model for U_{id} that can be used for simulation and analysis.

Hybrid system identification: double pendulum with soft contacts

We consider a hybrid linear system consisting in an inverted double pendulum with soft contacts at the joints, as depicted in Figure 7(a). This system has nine linear modes, depending on whether the contact force of each joint is inactive, active on the left or active on the right (Aydinoglu, Preciado, and Posa 2020). Our goal is to learn these linear modes as well as their domain of validity, from data. For that, we simulated trajectories and collected $N = 250$ sampled values of θ_1, θ_2 and the force applied on the lower joint. We used Algo. 4 to compute a PWA regression of the data with rectangular domains and with error tolerance $\varepsilon = 0.01$. The result is shown in Figure 7(b). The number of iterations of the algorithm was about 23,000 for a total time of 800 s.

We see that the affine pieces roughly divide the state space into a grid of 3×3 regions. This is consistent with our ground-truth model, in which the contact force at each joint has three linear modes depending only on the angle made at the joint. The PWA regression provided by Algo. 4 allows us to learn this feature of the system from data, without assuming anything about the system except that the domains of the affine pieces are rectangular.

We compare with SA regression and classical PWA regression. The MILP approach to solve the SA regression (Prob. 3) with $\varepsilon = 0.01$ and $M = 9$ could not handle more than 51 data points within reasonable time (1000 s); see Figure 8(a). Furthermore, the computed clusters of data points (see Figure 6) do not allow to learn a (simple) PWA model, thereby hindering to learn important features of the system. Last but not least, we compare with the recent tool PARC (Bemporad 2023).[6] The fitting accuracy on training data is high ($R^2 = 0.995$). The resulting partition of the input space is depicted in Figure 8(b). As we can see, PARC finds a PWA function with 8 modes, although an upper bound (K) of 10 was given. However, the regions do not align with the axis (as this is not enforced by the algorithm). Consequently, regions with a small number of samples (e.g., lower-right) are missing, while regions with many samples (e.g., central) are overly divided.

Hybrid system identification: carts with springs

We consider a hybrid linear system consisting in two carts with springs, as depicted in Figure 9(a). The force applied on the left cart has four linear modes, depending on the values of x_1 and x_2. Our

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

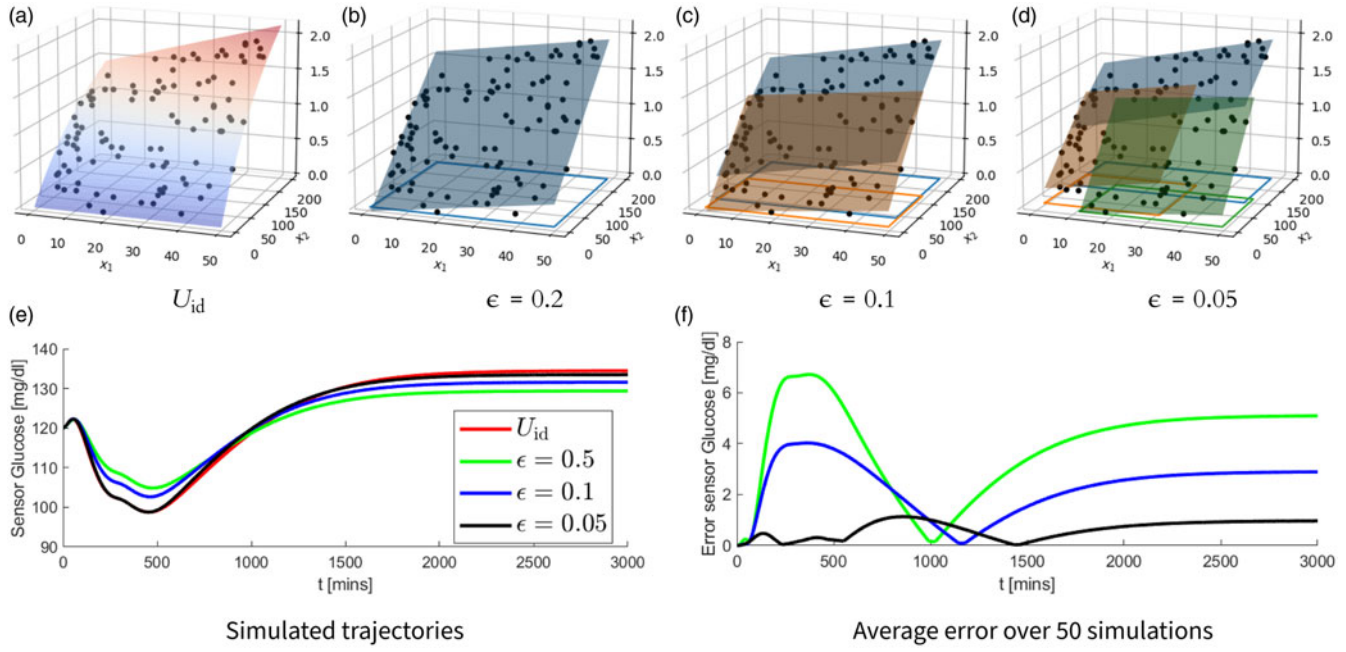


Figure 5. Glucose–insulin system. (a) 100 sampled points (black dots) on the graph of U_{id} (surface). (b,c,d) Optimal TPWA regression for various error tolerances ε. (e) Simulations using the nonlinear model versus the PWA approximations. (f) Error between nonlinear and PWA models averaged over 50 simulations with different initial conditions.

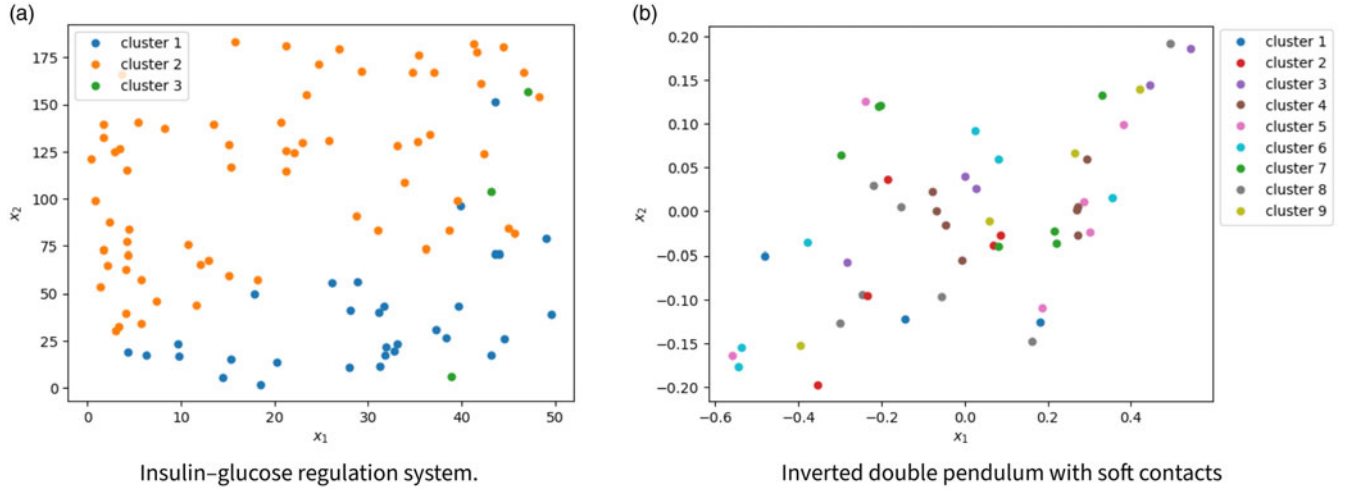


Figure 6. Clusters of data points from SA regression.

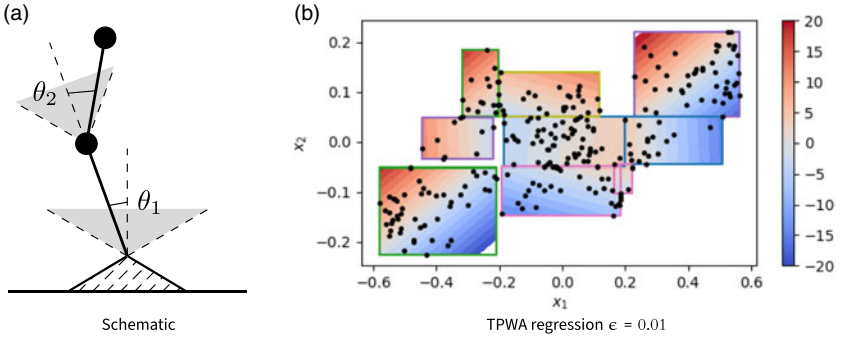


Figure 7. Inverted double pendulum with soft contacts. (a): Elastic contact forces apply when θ is outside gray region, (b): optimal TPWA regression of the data with rectangular domains.

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

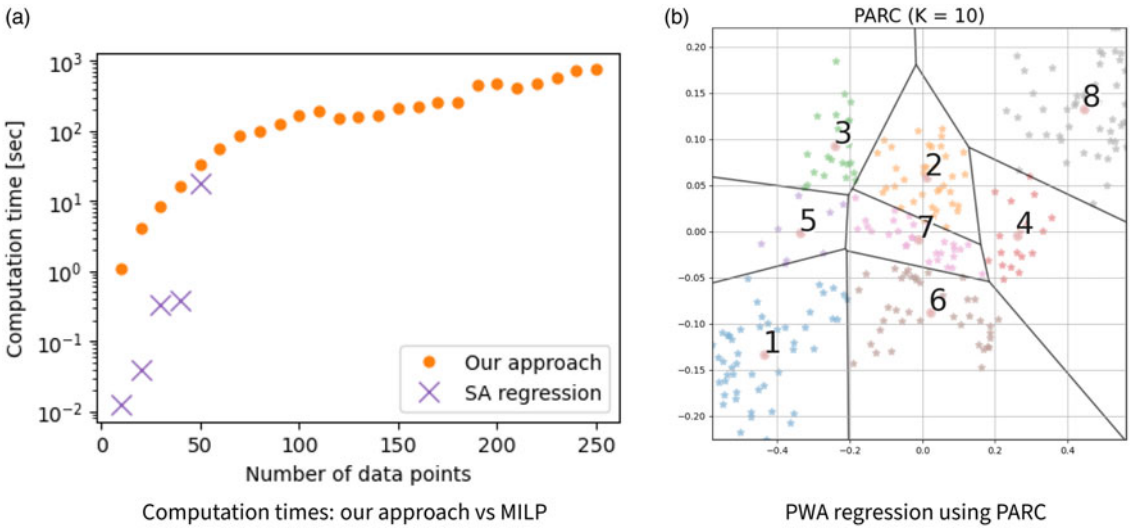


Figure 8. (a): Comparison with MILP approach for SA regression. Time limit is set to 1000 secs. (b) Partition of the input space by the PWA function computed using the state-of-the-art tool PARC (Bemporad 2023).

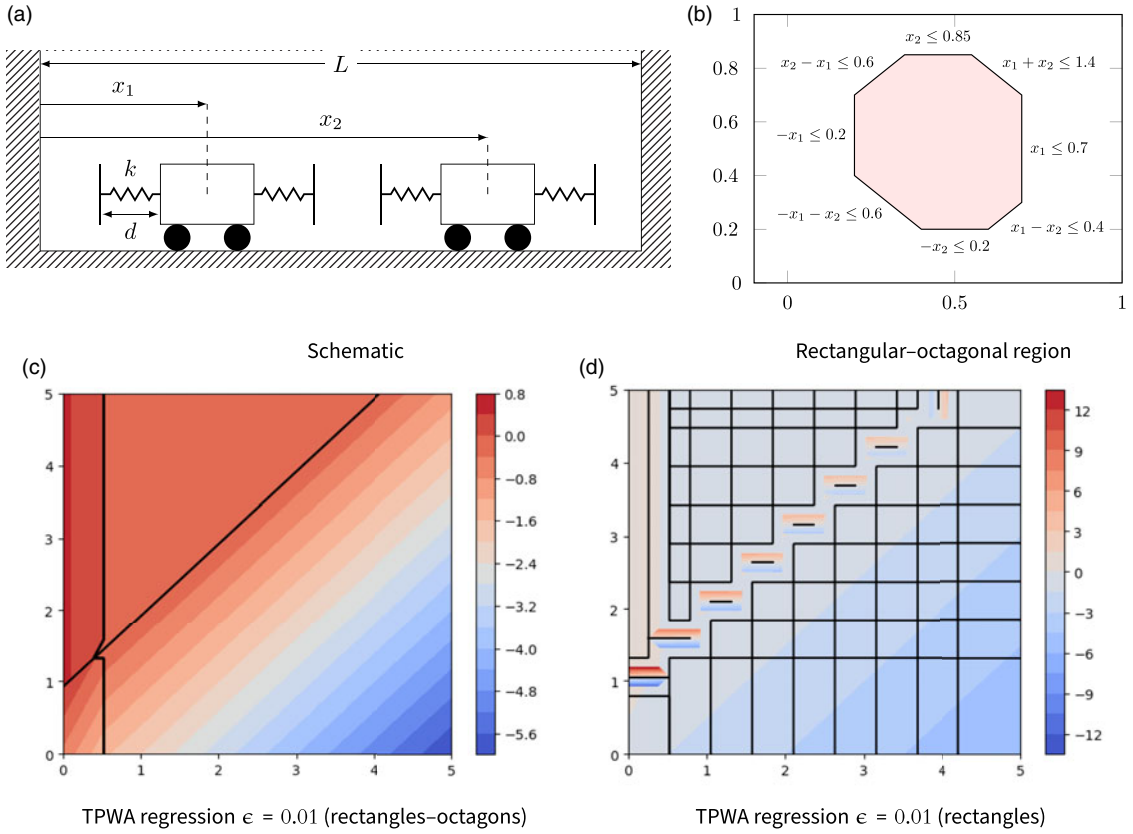


Figure 9. Carts with springs. (a): Elastic contact forces apply when the springs are compressed. (b): Example of rectangular–octagonal region. (c): Optimal TPWA regression of the data with rectangular–octagonal domains. (d): Optimal TPWA regression of the data with rectangular domains.

goal is to learn these linear modes as well as their domain of validity, from data. For that, we used $N = 400$ data obtained by gridding the input domain $[0, L]^2$ uniformly. We used Algo. 4 to compute a PWA regression with error tolerance $\varepsilon = 0.01$. We considered two templates: first the 'rectangular–octagonal' template wherein each region can have up to eight faces consisting in horizontal, vertical and oblique lines (see Figure 9(b) for an example); then, we compared with the rectangular template.

The results are shown in Figure 9(c,d). The running time of the algorithm was about 950 secs for the rectangular–octagonal template, and 120 s for the rectangular template. It is natural that the rectangular–octagonal template takes more time because we

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press

allow for degrees of freedom in the shape of the regions. However, we observe in Figure 9(c,d) that the most expressive template gives better result in terms of simplicity of the PWA function.

Conclusion

To conclude, we have presented an algorithm for fitting PWA models wherein each piece ranges over a region whose shape is dictated by a user-provided template. The complexity of the problem has been analyzed in terms of the number of data points, the dimension of the input domain and the template, X the desired number of pieces of the model. We have presented a top-down algorithm that explores subsets of the data guided by the concept of infeasibility certificates. Finally, our implementation provides some interesting applications of this approach to cyber-physical systems. Despite these contributions, the problem of identifying hybrid systems from data remains a computationally hard problem and the computational challenges of providing precise solutions with mathematical guarantees remain formidable. Our future work will investigate the use of better data structures to help scale our algorithms to larger and higher dimensional data sets. We are also investigating other approaches to PWA identification involving regions that are separated by arbitrary hyperplanes rather than fixed templates. Finally, we are interested in connections between the approach presented here and ideas from computational geometry. In particular, the link between the VC dimension of the shapes used to specify the regions in our PWA model and the complexity of the regression procedure offers interesting venues for future work.

Data availability statement. The code and data used in this paper are available at https://github.com/guberger/L4DC2023_Tool_PWA_Regression_Top-Down.

Author contribution. GB and SS conceived the method and the experimental setup, produced the technical results and wrote the paper together. GB implemented the code and produced the experimental results.

Financial support. This research was supported by grants from the Federation Wallonie-–Bruxelles (WBI) and the US National Science Foundation (NSF) under award numbers 1836900 and 1932189.

Competing interests. None.

Ethics statements. No experiments involving humans or animals were conducted in the context of this research.

Notes

1 See for instance the example in Berger, Narasimhamurthy, and Sankaranarayanan (2024, Lemma 2.5) for a system that can be described with the rectangular template but not as a guarded linear system.

2 This is a consequence of Farkas' Lemma and Carathéodory's Theorem; see Lemma 2.

3 In theory, by using Sauer–Shelah's lemma (see, e.g., Har-Peled 2011, Lemma 6.2.2), this number can be reduced to $\sum_{i=1}^{h} \binom{K}{i} \approx 4 \times 10^6$. This is because the *VC dimension* of rectangular regions in $\mathbb{R}^d$ is $2d$.

4 Note that L^1 regularization costs are often used in machine learning to induce sparsity of the optimal solution (Boyd and Vandenberghe 2004, p. 304). Here, we use a *weighted* L^1 regularization cost to induce a sparsity pattern dictated by the geometry of the problem.

5 The implementation is made in Julia, with Gurobi 11.0, under academic license, as LP and MILP solver (including for the optimal set cover problems). Our approach uses the standard set data structures available in Julia for manipulating index sets in order to implement the key steps of Algorithm 1. All computations were made on a laptop with processor Intel Core i7-7600u and 16GB RAM running Windows.

6 We used the default parameters proposed on the webpage https://github.com/bemporad/PyPARC.

Connections references

Paoletti N, Woodcock J (2023) How to ensure safety of learning-enabled cyber-physical systems? *Research Directions: Cyber-Physical Systems.* **1**, e2, 1–2. https://doi.org/10.1017/cbp.2023.2.

References

Aydinoglu A, Preciado VM and Posa M (2020) Contact-aware controller design for complementarity systems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 1525–1531. IEEE. https://doi.org/10.1109/ICRA40945.2020.9197568.

Bellman R. (1961) On the approximation of curves by line segments using dynamic programming. *Communications of the ACM* **4**(6), 284. https://doi.org/10.1145/366573.366611.

Bellman R and Roth R (1969) Curve fitting by segmented straight lines. *Journal of the American Statistical Association* **64**(327), 1079–1084. https://doi.org/10.1080/01621459.1969.10501038.

Bemporad A (2023) A piecewise linear regression and classification algorithm with application to learning and model predictive control of hybrid systems. *IEEE Transactions on Automatic Control* **68**(6):3194–3209. https://doi.org/10.1109/TAC.2022.3183036.

Bemporad A, Garulli A, Paoletti S and Vicino A (2005) A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control* **50**(10), 1567–1580. https://doi.org/10.1109/TAC.2005.856667.

Berger G, Narasimhamurthy M and Sankaranarayanan S (2024) Algorithms for identifying flagged and guarded linear systems. In *Proceedings of the 27th ACM International Conference on Hybrid Systems: Computation and Control*, 1–13. ACM. https://doi.org/10.1145/3641513.3650140.

Berger GO and Sankaranarayanan S (2023) Template-based piecewise affine regression. In Matni N, Morari M, and Pappas GJ (eds.), *Proceedings of the 5th Annual Learning for Dynamics and Control Conference*, 211, 509–520. Proceedings of Machine Learning Research.

Boyd S and Vandenberghe L (2004) *Convex Optimization*. Cambridge, UK: Cambridge University Press. https://doi.org/10.1017/CBO9780511804441.

Bradley PS and Mangasarian OL (2000) K-plane clustering. *Journal of Global optimization* **16**(1), 23–32. https://doi.org/10.1023/A:1008324625522.

Breiman L (1993) Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory* **39**(3), 999–1013. https://doi.org/10.1109/18.256506.

Chvatal V (1979) A greedy heuristic for the set-covering problem. *Mathematics of Operations Research* **4**(3), 233–235. https://doi.org/10.1287/moor.4.3.233.

Dalla Man C, Rizza RA and Cobelli C (2007) Meal simulation model of the glucose-insulin system. *IEEE Transactions on Biomedical Engineering* **54**(10), 1740–1749. https://doi.org/10.1109/TBME.2007.893506.

Ferrari-Trecate G, Muselli M, Liberati D and Morari M (2003) A clustering technique for the identification of piecewise affine systems. *Automatica* **39**(2), 205–217. https://doi.org/10.1016/S0005-1098(02)00224-8.

Har-Peled S (2011) *Geometric Approximation Algorithms*. Vol. 173. Mathematical Surveys and Monographs. Providence, RI: American Mathematical Society.

Jin W, Aydinoglu A, Halm M and Posa M (2022) Learning linear complementarity systems. In Firoozi R, Mehr N, Yel E, Antonova R, Bohg J, Schwager M and Kochenderfer M (eds.), *Proceedings of the 4th Annual Learning for Dynamics and Control Conference* 168, 1137–1149. Proceedings of Machine Learning Research.

Jungers RM (2009) *The Joint Spectral Radius: Theory and Applications*. Berlin: Springer. https://doi.org/10.1007/978-3-540-95980-9.

Lauer F (2013) Estimating the probability of success of a simple algorithm for switched linear regression. *Nonlinear Analysis: Hybrid Systems* **8**, 31–47. https://doi.org/10.1016/j.nahs.2012.10.001.

Lauer F and Bloch G (2019) *Hybrid System Identification: Theory and Algorithms for Learning Switching Models*. Cham: Springer. https://doi.org/10.1007/978-3-030-00193-3.

Medhat R, Ramesh S, Bonakdarpour B and Fischmeister S (2015) A framework for mining hybrid automata from input/output traces. In *2015 International Conference on Embedded Software (Emsoft)*, 177–186. IEEE. https://doi.org/10.1109/EMSOFT.2015.7318273.

Miné A (2006) The octagon abstract domain. *Higher-Order and Symbolic Computation* **19**(1), 31–100. https://doi.org/10.1007/s10990-006-8609-1.

Münz E and V Krebs (2002) Identification of hybrid systems using a priori knowledge. *IFAC Proceedings Volumes* **35**(1), 451–456. https://doi.org/10.3182/20020721-6-ES-1901.00563.

Münz E and Krebs V (2005) Continuous optimization approaches to the identification of piecewise affine systems. *IFAC Proceedings* **38**(1), 349–354. https://doi.org/10.3182/20050703-6-CZ-1902.00342.

Nakada H, Takaba K and Katayama T (2005) Identification of piecewise affine systems based on statistical clustering technique. *Automatica* **41**(5), 905–913. https://doi.org/10.1016/j.automatica.2004.12.005.

Ozay N (2016) An exact and efficient algorithm for segmentation of ARX models. In *2016 American Control Conference (ACC)*, 38–41. IEEE. https://doi.org/10.1109/ACC.2016.7524888.

Ozay N, Lagoa C and Sznaier M (2009) Robust identification of switched affine systems via moments-based convex optimization. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) Held Jointly with 2009 28th Chinese Control Conference*, 4686–4691. IEEE. https://doi.org/10.1109/CDC.2009.5399962.

Ozay N, Sznaier M, Lagoa CM and Camps OI (2012) A sparsification approach to set membership identification of switched affine systems. *IEEE Transactions on Automatic Control* **57**(3), 634–648. https://doi.org/10.1109/TAC.2011.2166295.

Paoletti S, Juloski AL, Ferrari-Trecate G and Vidal R (2007) Identification of hybrid systems — a tutorial. *European Journal of Control* **13**(2–3), 242–260. https://doi.org/10.3166/ejc.13.242-260.

Pavlidis T and Horowitz SL (1974) Segmentation of plane curves. *IEEE Transactions on Computers* **63**(8), 860–870. https://doi.org/10.1109/T-C.1974.224041.

Porreca R, Drulhe S, de Jong H, and Ferrari-Trecate G (2009) Identification of parameters and structure of piecewise affine models of genetic networks. *IFAC Proceedings Volumes* **42**(10), 587–592. https://doi.org/10.3182/20090706-3-FR-2004.00097.

Rockafellar R (1970) *Convex Analysis*. Princeton, NJ: Princeton University Press.

Sadraddini S and Belta C (2018) Formal guarantees in data-driven model identification and control synthesis. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control*, 147–156. ACM. https://doi.org/10.1145/3178126.3178145.

Smarra F, Di Girolamo GD, De Iuliis V, Jain A, Mangharam R and D'Innocenzo A (2020) Data-driven switchingmodeling formpc using regression trees and random forests. *Nonlinear Analysis: Hybrid Systems* **36** (100882), 1–20. https://doi.org/10.1016/j.nahs.2020.100882.

Soto MG, Henzinger TA and Schilling C (2021) Synthesis of hybrid automata with affine dynamics from time-series data. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, 1–11. ACM. https://doi.org/10.1145/3447928.3456704.

Vanderbei RJ (2020) *Linear Programming: Foundations and Extensions*. 5th ed. Cham: Springer. https://doi.org/10.1007/978-1-4614-7630-6.

Vidal R, Soatto S, Ma Y and Sastry S (2003) An algebraic geometric approach to the identifica669 tion of a class of linear hybrid systems. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03ch37475)*, 167–172. IEEE. https://doi.org/10.1109/CDC.2003.1272554.

Yuan Y, Tang X, Zhou W, Pan W, Li X, Zhang H-T, Ding H and Goncalves J (2019) Data driven discovery of cyber physical systems. *Nature Communications* **10**(4894), 1–9. https://doi.org/10.1038/s41467-019-12490-1.

https://doi.org/10.1017/cbp.2024.3 Published online by Cambridge University Press