Automatica 117 (2020) 109002

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Rao-Blackwellized sampling for batch and recursive Bayesian inference of Piecewise Affine models☆

Dario Piga[a,*], Alberto Bemporad[b], Alessio Benavoli[c]

[a] Dalle Molle Institute for Artificial Intelligence Research - USI/SUPSI, Galleria 2, Via Cantonale 2c, CH-6928 Manno, Switzerland
[b] IMT School for Advanced Studies Lucca, Piazza San Francesco 19, 55100 Lucca, Italy
[c] University of Limerick, 13657 Limerick, Ireland

ARTICLE INFO

Article history:
Received 22 April 2019
Received in revised form 2 February 2020
Accepted 5 April 2020
Available online xxxx

Keywords:
Piecewise-affine regression
Bayesian inference
Markov Chain Monte Carlo
Recursive identification
Particle filters

ABSTRACT

This paper addresses batch (offline) and recursive (online) Bayesian inference of *Piecewise Affine* (PWA) regression models. By exploiting the particular structure of PWA models, efficient Rao-Blackwellized Monte Carlo sampling algorithms are developed to approximate the joint posterior distribution of the model parameters. Only the marginal posterior of the parameters used to describe the regressor-space partition is approximated, either in a batch mode using a Metropolis–Hastings *Markov-Chain Monte Carlo* (MCMC) sampler, or sequentially using *particle filters*, while the conditional distribution of the other model parameters is computed analytically. Probability distributions for the predicted outputs given new test inputs are derived and modifications of the proposed approaches to address *maximum-a-posteriori* estimate are discussed. The performance of the proposed algorithms is shown via a numerical example and through a benchmark case study on data-driven modelling of the electronic component placement process in a pick-and-place machine.

1. Introduction

1.1. PieceWise Affine modelling

PieceWise Affine (PWA) model formalism provides a powerful and flexible tool to describe complex nonlinear regressor-to-output mappings as the collection of simple affine submodels, each associated to a polyhedral region of the regressor domain. They can be then used to describe systems which change their dynamics due, for example, to: saturations, thresholds, dead-zones, abrupt changes of the working environment (like manipulators which alternate between free and contact motion), etc.

Thanks to their universal approximation property, PWA maps are able to approximate any sufficiently smooth nonlinear function with arbitrary accuracy (Breiman, 1993). Moreover, because of the equivalence between PWA and hybrid linear models (Heemels, De Schutter, & Bemporad, 2001), well settled tools for modelling, analysis and control of hybrid systems can be applied to systems represented in a PWA form (Bemporad, Ferrari-Trecate, & Morari, 2000; Bemporad & Morari, 1999).

1.2. Algorithms for PWA regression

Learning PWA models from regressor/output data is an NP-hard problem (Lauer, 2015), which needs to estimate both the parameters defining the local affine models and the partition of the regressor space. Several algorithms/heuristics have been developed in the last years for PWA regression or, more in general, for data-driven modelling of hybrid systems, which are characterized by the interaction between discrete (logic) and continuous (physical) states.

Multi-stage clustering-based approaches are proposed in Bako, Boukharouba, Duviella, and Lecoeuche (2011), Bemporad, Garulli, Paoletti, and Vicino (2005), Breschi, Piga, and Bemporad (2016), Ferrari-Trecate, Muselli, Liberati, and Morari (2003), Juloski, Weiland, and Heemels (2005) and Naik, Mejari, Piga, and Bemporad (2017). The main idea behind these methods is to first cluster the training regressors according to a certain criterion and then estimate the parameters of the local affine functions using standard methods for identification of linear systems (*e.g.*, least squares). In Ferrari-Trecate et al. (2003), the regressors are clustered using a *k*-means algorithm and the parameters of the local affine maps are estimated through weighted least squares. The approaches in Bako et al. (2011), Breschi et al. (2016), Juloski et al. (2005) and Naik et al. (2017) are based on a *greedy* strategy where training data is processed sequentially. At each iteration, clustering is performed by assigning the current regressor to the local model that

☆ The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Adrian George Wills under the direction of Editor Torsten Söderström.

[*] Corresponding author.

E-mail addresses: dario.piga@supsi.ch (D. Piga), alberto.bemporad@imtlucca.it (A. Bemporad), alessio.benavoli@ul.ie (A. Benavoli).

https://doi.org/10.1016/j.automatica.2020.109002
0005-1098/© 2020 Elsevier Ltd. All rights reserved.

"best describes" the current regressor-output sample. The parameters of this submodel are simultaneously updated via recursive least squares in Bako et al. (2011), Breschi et al. (2016) and Naik et al. (2017), and through particle approximation in Juloski et al. (2005). Least squares are used at a second stage to estimate the parameters of the local affine maps. In Bemporad et al. (2005), PWA regression is formulated in a bounded-error identification framework. Clustering, parameter identification, and estimation of the number of local affine submodels are performed simultaneously by partitioning a suitable set of linear complementary inequalities into a minimum number of feasible subsystems. All the approaches in Bako et al. (2011), Bemporad et al. (2005), Breschi et al. (2016), Ferrari-Trecate et al. (2003) and Juloski et al. (2005) compute the polyhedral partition of the regressor space at a second stage, once the regressors are clustered.

Optimization-based approaches are proposed in Bako (2011), Ohlsson and Ljung (2013), Piga and Tóth (2013) and Roll, Bemporad, and Ljung (2004). Piecewise affine regression is formulated in Roll et al. (2004) as a mixed-integer linear or quadratic programming problem and solved by *branch-and-bound*. The contributions (Bako, 2011; Ohlsson & Ljung, 2013; Piga & Tóth, 2013) address identification of switching systems and formulate an over-parametrized least-squares problem with a LASSO-like regularization term penalizing the number of switches.

1.3. Paper contribution

In this work, PWA regression is addressed in a Bayesian setting, deriving the posterior distribution of the model parameters and of the predicted output. Efficient Rao-Blackwellized sampling algorithms tailored for PWA regression are developed to approximate the posterior distribution of the parameters characterizing the PWA model. More specifically, the following two problems are addressed:

- *batch* (offline) learning through Rao-Blackwellized Metropolis–Hastings *Markov-Chain Monte Carlo* (MCMC) sampling;
- *recursive* (online) learning through Rao-Blackwellized *particle filters*.

By exploiting the peculiar structure of PWA functions, only the marginal posterior of the parameters used to define the regressor-space partition is approximated through MCMC simulation or particle filters, while the conditional posterior distribution of the other parameters (given the regressor-space partition) is computed analytically. Modifications of the proposed algorithms to address both batch and recursive *Maximum-A-Posteriori* (MAP) estimates are also discussed.

For the sake of completeness, it is worth mentioning that Markov-Chain Monte Carlo algorithms have been already employed in Pillonetto (2016) and Wågberg, , Lindsten, and Schön (2015) for batch PWA regression. Both Pillonetto (2016) and Wågberg et al. (2015) employ a Gibbs sampler to approximate the marginal distribution of the whole sequence of the active local submodels in the training set. As a consequence, the sampling space increases with the number of training data. On the other hand, the dimension of the space sampled by the Rao-Blackwellized algorithms proposed in this paper does not depend on the size of the training set.

Another important difference with respect to Pillonetto (2016) is that our approach estimates the local submodels and the partition of the regressor domain in one shot, while Pillonetto (2016) proposes a two-step procedure tailored to identification of hybrid dynamical systems. Specifically, Gibbs sampling is used at the first stage to compute the sequence of active submodels through maximum likelihood. This information is used at a second stage to estimate the affine submodels via *stable spline kernels* and to partition the regressor space via linear separations methods. The advantage in using stable spline kernels is that the regressor (implicitly described in Pillonetto (2016) by the order of the linear dynamical submodels and past inputs) is not a-priori specified. On the other hand, the approach in the present paper does not address automated feature selection and, when applied to the identification of dynamical systems, the order of the local submodels has to be specified a priori.

1.4. Paper outline

The paper is organized as follows. After formally introducing PWA models and prior modelling assumptions, the problem of Bayesian inference for PWA models is formulated in Section 2. Batch learning is discussed in Section 3, where the developed Rao-Blackwellized Metropolis–Hastings MCMC algorithm tailored for PWA regression is presented. Recursive learning through Rao-Blackwellized particle filters is presented in Section 4. The effectiveness of the proposed algorithms is illustrated in Section 5 by means of a numerical example and a benchmark case study. Conclusions are drawn in Section 6. Detailed proofs of the main results of the paper are reported in the Appendix.

1.5. Notation

The following notation is used throughout the paper. Let $\mathbb{R}^+$ be the set of positive real numbers, $\mathbb{R}^n$ be the set of real vectors of dimension n, $\mathbb{R}^{n,m}$ be the set of real matrices with n rows and m columns, and I_n be the identity matrix of size n. Given a matrix $A \in \mathbb{R}^{n,m}$, A' and $A^{(j)}$ denote the transpose and the jth column of A, respectively. If A is square, $|A|$ denotes its determinant.

The Dirac delta function centred at $\bar{x}$ is denoted by $\delta_{\bar{x}}(x)$ and $\mathbb{I}$ denotes the indicator function defined, for a given statement S, as $\mathbb{I}(S) = 1$ is S is true, 0 otherwise.

Given two positive parameters $\alpha, \beta \in \mathbb{R}^+$, $\Gamma(x; \alpha, \beta)$ denotes the *probability density function* of a *Gamma*-distributed positive random variable $x \in \mathbb{R}^+$, *i.e.*, $\Gamma(x; \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$, where $\Gamma(\alpha)$ is the *Gamma* function evaluated at α. For a random matrix $A \in \mathbb{R}^{n,m}$, we refer to $p(A)$ as the probability distribution of $vec(A)$, where $vec(A) \in \mathbb{R}^{nm}$ is the vector obtained by stacking the columns of A on top of one another. Thus, when referring to the covariance matrix of A, we mean the covariance matrix of the vector $vec(A)$.

2. Problem formulation

2.1. PWA model

Consider a training set of inputs $X = \{x_t\}_{t=1}^T$ and outputs $Y = \{y_t\}_{t=1}^T$, where t denotes the index (*e.g.*, time) of the data sequence, $x_t \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the *regressor* or *input* and $y_t \in \mathcal{Y} \subseteq \mathbb{R}^{n_y}$ is the *output*. The observation model is:

$$y_t = f(x_t) + v_t, \tag{1a}$$

where $v_t \in \mathbb{R}^{n_y}$ is a multivariate zero-mean white Gaussian noise statistically independent of the input x_t. For clarity of exposition and in order not to heavy the notation, we assume a diagonal noise covariance matrix $\sigma_v^2 I_{n_y}$, with $\sigma_v^2 \in \mathbb{R}^+$. This corresponds to the assumption that the noises on each output channel are statistically independent of each other and they share the same variance σ_v^2.

The function $f:\mathcal{X}\to\mathcal{Y}$ is PWA and defined by

$$f(x_t)=\begin{cases}\theta_1'\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right] & \text{if } x_t\in\mathcal{X}_1,\\ \vdots & \\ \theta_s'\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right] & \text{if } x_t\in\mathcal{X}_s,\end{cases} \tag{1b}$$

where $\theta_i\in\mathbb{R}^{n_\theta,n_y}$, with $n_\theta=n_x+1$ and $i=1,\dots,s$, are used to parametrize the local affine functions; $s\in\mathbb{N}$ denotes the number of modes (*i.e.*, number of affine functions $\theta_i'\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]$ describing f) and $\mathcal{X}_i\subseteq\mathcal{X}$, $i=1,\dots,s$, are polyhedra that form a complete polyhedral partition[1] of the regressor domain $\mathcal{X}$. We further stress that, based on the definition (1b), the function f is locally affine in the region $\mathcal{X}_i$ and it has the form $f(x_t)=\theta_i'\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]$. We assume that the polyhedra $\mathcal{X}_i$ are generated by a piecewise-linear separator function (Bennett & Mangasarian, 1994) and are defined in terms of $s-1$ linear inequalities as

$$\mathcal{X}_i=\left\{x\in\mathbb{R}^{n_x}:\omega_i'\left[\begin{smallmatrix}1\\x\end{smallmatrix}\right]\geq\omega_j'\left[\begin{smallmatrix}1\\x\end{smallmatrix}\right],\ j=1,\dots,s, j\neq i\right\}. \tag{2}$$

The vectors $\omega_i\in\mathbb{R}^{n_\omega}$ (with $n_\omega=n_\theta=n_x+1$ and $i=1,\dots,s$) define the separator function $\phi(x)=\max_{i=1,\dots,s}\omega_i'\left[\begin{smallmatrix}1\\x\end{smallmatrix}\right]$ and are used to parametrize $\mathcal{X}_i$ in (2).

2.2. Learning problem

The PWA regression function f in (1) is described by the following parameters:

- s, the number of modes;
- $\Theta=[\theta_1\ \dots\ \theta_s]\in\mathbb{R}^{n_\theta,n_ys}$, the collection of parameters defining the local affine functions in (1b);
- $\Omega=[\omega_1\ \dots\ \omega_s]\in\mathbb{R}^{n_\omega,s}$, the collection of parameters defining the polyhedra $\{\mathcal{X}_i\}_{i=1}^s$ according to the linear-inequality representation (2). In the rest of the paper, the partition of the regressor space $\mathcal{X}$ generated by the parameters Ω is denoted as $\mathcal{X}[\Omega]$, and $\{\mathcal{X}_i[\Omega]\}_{i=1}^s$ denotes the corresponding polyhedral regions. To simplify the notation, the dependence of $\mathcal{X}_i$ on Ω will be stressed only if needed;
- σ_v^2, the variance of the noise on each output channel. In the rest of the paper, model (1) is parametrized as a function of the noise precision σ_v^{-2}.

In this work, the number of modes s is fixed a priori and the object of interest is the posterior distribution $p(\Theta,\Omega,\sigma_v^{-2}|X,Y)$ of the parameters given the training data X,Y. The posterior distribution can be then used, for instance, to make point or interval predictions on the output $y^\star$ for a new test input $x^\star$. In case the parameter s is not known a priori, s can be chosen by cross validation, with an upper-bound s_{max} dictated by the maximum tolerated complexity of the PWA function f.

2.3. Priors over the parameters

In order to compute the posterior distribution of the parameters $\Theta,\Omega,\sigma_v^{-2}$, the following priors are assumed.

A1. The parameters ω_i, $i=1,\dots,s$, follow a zero-mean Gaussian distribution with covariance matrix $\sigma_\omega^2I_{n_\omega}$, *i.e.*,

$$\omega_i\sim\ p(\omega_i)=\mathcal{N}(\omega_i;0,\sigma_\omega^2I_{n_\omega}), \tag{3}$$

where $\sigma_\omega>0$ is a hyper-parameter characterizing the prior on ω_i. Furthermore, the parameters Ω are assumed to be mutually independent, *i.e.*, $\Omega\sim\ p(\Omega)=\prod_{i=1}^s p(\omega_i)$.

A2. The parameters $\Theta,\Omega,\sigma_v^{-2}$ are statistically independent of the input data X. Furthermore, Ω is statistically independent of Θ and σ_v^{-2}, *i.e.*,

$$p(\Theta,\Omega,\sigma_v^{-2}|X)=p(\Theta,\Omega,\sigma_v^{-2})=p(\Theta,\sigma_v^{-2})p(\Omega).$$

A3. The joint prior probability distribution of θ_i and σ_v^{-2} is a *Normal-Gamma* which factorizes as $p(\theta_i^{(j)},\sigma_v^{-2})=p(\theta_i^{(j)}|\sigma_v^{-2})p(\sigma_v^{-2})$, with

$$p(\theta_i^{(j)}|\sigma_v^{-2})=\mathcal{N}\left(\theta_i^{(j)};0,\frac{1}{\sigma_v^{-2}}\lambda^2I_{n_\theta}\right), \tag{4a}$$

$$p(\sigma_v^{-2})=\Gamma\left(\sigma_v^{-2};\alpha_0,\beta_0\right), \tag{4b}$$

where $\lambda>0$ is a hyper-parameter characterizing the prior distribution $p(\theta_i^{(j)}|\sigma_v^{-2})$, and $\theta_i^{(j)}$, $j=1,\dots,n_y$, denotes the jth column of matrix θ_i (namely, the set of parameters θ_i characterizing the jth output $y_t^{(j)}$ of the ith local model). Furthermore, for simplicity, the parameters Θ are assumed to be mutually independent given σ_v^{-2}, *i.e.*,

$$\begin{aligned}\Theta|\sigma_v^{-2}\sim p(\Theta|\sigma_v^{-2})&=\prod_{i=1}^s\prod_{j=1}^{n_y}p(\theta_i^{(j)}|\sigma_v^{-2})\\&=\frac{(\lambda^{-2}\sigma_v^{-2})^{\frac{n_yn_\theta s}{2}}}{(2\pi)^{\frac{n_yn_\theta s}{2}}}\prod_{i=1}^s\prod_{j=1}^{n_y}e^{-\frac{1}{2}\lambda^{-2}\sigma_v^{-2}\theta_i^{(j)'}\theta_i^{(j)}}.\end{aligned} \tag{5}$$

A Normal-Gamma prior is assumed in [A3] as it represents the conjugate prior of a Gaussian likelihood with unknown mean and variance (Bishop, 2006, Ch. 2). This choice will allow us to obtain an analytical expression for the conditional posterior of Θ and σ_v^{-2} given the parameters Ω.

2.4. Posterior distribution

Using Bayes' rule, $p(\Theta,\Omega,\sigma_v^{-2}|X,Y)$ is given by:

$$p(\Theta,\Omega,\sigma_v^{-2}|X,Y)=\frac{p(Y|\Theta,\Omega,\sigma_v^{-2},X)p(\Theta,\Omega,\sigma_v^{-2})}{p(Y|X)}. \tag{6}$$

Since the noise samples v_t in (1a) are assumed to be independent and identically distributed, the *likelihood* $p(Y|\Theta,\Omega,\sigma_v^{-2},X)$ is thus given by

$$\begin{aligned}p(Y|\Theta,\Omega,\sigma_v^{-2},X)&=\prod_{t=1}^T p(y_t|\Theta,\Omega,\sigma_v^{-2},x_t)\\&=\mathcal{N}\left(y_t;\theta_{s_t}'\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right],\frac{1}{\sigma_v^{-2}}I_{n_y}\right),\end{aligned} \tag{7}$$

where s_t denotes the active mode at index t, *i.e.*, $s_t=i\Leftrightarrow x_t\in\mathcal{X}_i[\Omega]$.

Borrowing the notation from Pillonetto (2016), the likelihood (7) can be written as

$$\begin{aligned}&p(Y|\Theta,\Omega,\sigma_v^{-2},X)\\&=\frac{(\sigma_v^{-2})^{\frac{n_yT}{2}}}{(2\pi)^{\frac{n_yT}{2}}}\prod_{i=1}^s\prod_{j=1}^{n_y}e^{-\frac{1}{2}\sigma_v^{-2}\left(\mathbb{Y}_i^{(j)}-\mathbb{X}_i'\theta_i^{(j)}\right)'\left(\mathbb{Y}_i^{(j)}-\mathbb{X}_i'\theta_i^{(j)}\right)},\end{aligned} \tag{8}$$

where $\mathbb{Y}_i^{(j)}$, $i=1,\dots,s$ and $j=1,\dots,n_y$, is the column vector associated to the ith mode and to the jth output channel, whose components are taken from the sequence $\{y_t\}_{t=1}^T$ as follows: y_t' is a row of $\mathbb{Y}_i\Leftrightarrow s_t=i$, with $\mathbb{Y}_i=[\mathbb{Y}_i^{(1)}\ \dots\ \mathbb{Y}_i^{(n_y)}]$. The regressor matrix $\mathbb{X}_i$ is constructed accordingly, *i.e.*,

$$\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]=k\text{th column of }\mathbb{X}_i\Leftrightarrow y_t=k\text{th row of }\mathbb{Y}_i. \tag{9}$$

In other words, $\mathbb{Y}_i$ and $\mathbb{X}_i$ are constructed by stacking all and only output and input samples associated to mode i.

[1] $\{\mathcal{X}_i\}_{i=1}^s$ is a complete partition of $\mathcal{X}$ if $\bigcup_{i=1}^s\mathcal{X}_i=\mathcal{X}$ and $\mathring{\mathcal{X}}_i\cap\mathring{\mathcal{X}}_j=\emptyset$, $\forall i\neq j$, where $\mathring{\mathcal{X}}_i$ denotes the interior of $\mathcal{X}_i$.

Note that the matrices $\mathbb{Y}_i$, and consequently $\mathbb{X}_i$, depend on the partition of the regressor space.

The following sections present the developed Rao-Blackwellized sampling algorithms, tailored to PWA models, to compute (an approximation of) the posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X, Y)$ both in a batch mode (Section 3) and recursively (Section 4).

3. Rao-Blackwellised Metropolis–Hastings MCMC for batch learning

When batch learning is addressed, the posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X, Y)$ can be approximated through MCMC algorithms, which attempt to simulate draws from a complex distribution of interest (Andrieu, de Freitas, Doucet, & Jordan, 2003).

A *naive* application of MCMC to the considered PWA regression problem consists in generating a sequence of M random samples $\Theta[k]$, $\Omega[k]$, $\sigma_v^{-2}[k]$, $k = 1, 2, \ldots, M$, from an irreducible and aperiodic Markov chain whose stationary distribution is the target posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X, Y)$. The posterior is then approximated with the empirical point-mass distribution

$$p(\Theta, \Omega, \sigma_v^{-2}|X, Y) \approx \frac{1}{M}\sum_{k=1}^{M} \delta_{(\Theta[k], \Omega[k], \sigma_v^{-2}[k])}(\Theta, \Omega, \sigma_v^{-2}).$$

3.1. Rao-Blackwellised approach

Instead of using *naive* MCMC methods, which would require to draw samples from the high-dimensional parameter space $(\Theta, \Omega, \sigma_v^{-2})$, a hybrid approach is exploited, where part of the posterior is computed analytically and the other part is approximated using an MCMC sampler. Specifically, after factorizing the joint posterior distribution as

$$p(\Theta, \Omega, \sigma_v^{-2}|X, Y) = p(\Theta, \sigma_v^{-2}|\Omega, X, Y)p(\Omega|X, Y),$$

the structure of PWA models is exploited to compute an analytical expression for $p(\Theta, \sigma_v^{-2}|\Omega, X, Y)$, while the marginal posterior $p(\Omega|X, Y)$ is approximated with the point mass distribution

$$p(\Omega|X, Y) = \frac{1}{M}\sum_{k=1}^{M} \delta_{\Omega[k]}(\Omega) \tag{10}$$

through MCMC simulation, thus avoiding sampling over the parameter space (Θ, σ_v^{-2}). Summarizing, the posterior will be finally approximated by

$$p(\Theta, \Omega, \sigma_v^{-2}|X, Y) \approx p(\Theta, \sigma_v^{-2}|\Omega, X, Y)\frac{1}{M}\sum_{k=1}^{M} \delta_{\Omega[k]}(\Omega). \tag{11}$$

Approximating only a marginal of the distribution of interest in Monte Carlo sampling methods (such as MCMC, importance sampling, or particle filtering) is commonly referred to as *Rao-Blackwellised* approach, and has the advantage of reducing the variance of Monte Carlo estimates (Casella & Robert, 1996).

The following proposition provides the analytical expression for the conditional distribution $p(\Theta, \sigma_v^{-2}|\Omega, X, Y)$. The computation of the marginal $p(\Omega|X, Y)$ through Metropolis–Hastings MCMC is discussed in Section 3.2.

Proposition 1. *The posterior conditional distribution $p(\Theta, \sigma_v^{-2}|\Omega, X, Y)$ is a Normal-Gamma given by*

$$p(\Theta, \sigma_v^{-2}|\Omega, X, Y) = \underbrace{\Gamma(\sigma_v^{-2}; \alpha, \beta)}_{p(\sigma_v^{-2}|\Omega, X, Y)} \underbrace{\prod_{i=1}^{s}\prod_{j=1}^{n_y} \mathcal{N}(\theta_i^{(j)}; \mu_i^{(j)}, \sigma_v^2 F_i)}_{p(\Theta|\sigma_v^{-2}, \Omega, X, Y)}, \tag{12a}$$

with

$$F_i = \left(\mathbb{X}_i\mathbb{X}_i' + \lambda^{-2} I_{n_\theta}\right)^{-1}, \tag{12b}$$

$$\mu_i = \left(\mathbb{X}_i\mathbb{X}_i' + \lambda^{-2} I_{n_\theta}\right)^{-1} \mathbb{X}_i\mathbb{Y}_i = F_i\mathbb{X}_i\mathbb{Y}_i, \tag{12c}$$

$$\alpha = \alpha_0 + \frac{n_y T}{2}, \tag{12d}$$

$$\beta = \beta_0 + \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}\left(\mathbb{Y}_i^{(j)'}\mathbb{Y}_i^{(j)} - \mathbb{Y}_i^{(j)'}\mathbb{X}_i'\mu_i^{(j)}\right). \tag{12e}$$

Proposition 1 follows because of conjugacy between likelihood and prior. A detailed proof is in Appendix A.1.

3.2. Approximation of $p(\Omega|X, Y)$ through MCMC

Let us now focus on the approximation of the marginal posterior $p(\Omega|X, Y)$ through MCMC simulation. The well known Metropolis–Hastings MCMC algorithm (Chib & Greenberg, 1995) is used to draw samples from $p(\Omega|X, Y)$.

The Metropolis–Hastings MCMC sampler is reviewed in Algorithm 1. The algorithm simulates a Markov Chain with stationary distribution $p(\Omega|X, Y)$, and it requires to specify: an initial sample $\Omega[0]$; a random-walk proposal distribution $q(\Omega^*|\Omega[k])$; the length of the Markov Chain (namely, number of iterations) M. At each iteration k, a proposal Ω^* is drawn from the distribution $q(\Omega^*|\Omega[k])$ (Step 1.1) and accepted with probability $\mathcal{A}(\Omega^*, \Omega[k])$ (Steps 1.2 and 1.3). If the proposal Ω^* is accepted then $\Omega[k+1]$ is set to Ω^* (Step 1.4), otherwise $\Omega[k+1]$ is set to $\Omega[k]$ (Step 1.5). The output is the sequence of samples $\{\Omega[k]\}_{k=1}^{M}$ generated during the execution of the algorithm.

Implementing Algorithm 1 only requires the acceptance probability $\mathcal{A}(\Omega^*, \Omega[k])$ to be computed (Step 1.2). Since the proposal is chosen by the user, the only challenge in evaluating $\mathcal{A}(\Omega^*, \Omega[k])$ is to compute $\frac{p(\Omega^*|X, Y)}{p(\Omega[k]|X, Y)}$, whose value is given by the following proposition.

Proposition 2. *For given Ω^* and $\Omega[k]$, the ratio $\frac{p(\Omega^*|X, Y)}{p(\Omega[k]|X, Y)}$ is equal to*

$$\frac{(\beta[k])^{\alpha}}{(\beta^*)^{\alpha}} \frac{\prod_{i=1}^{s}|\mathbb{X}_i^*(\mathbb{X}_i^*)' + \lambda^{-2}I_{n_\theta}|^{-\frac{n_y}{2}}}{\prod_{i=1}^{s}|\mathbb{X}_i[k]\mathbb{X}_i'[k] + \lambda^{-2}I_{n_\theta}|^{-\frac{n_y}{2}}} \frac{p(\Omega^*)}{p(\Omega[k])}, \tag{13}$$

where $p(\Omega)$ is the prior on Ω given in (3), and $\mathbb{X}_i[k]$ (resp. $\beta[k]$) are defined as in (9) (resp. (12e)) based on the partition $\mathcal{X}[\Omega[k]]$.[2]

See Appendix A.2 for a proof of Proposition 2.

Remark 1. The proposal distribution $q(\Omega^*|\Omega[k])$ is the main tuning parameter in MCMC simulation. Indeed, for small-variance proposal distributions the proposal samples move around the space slowly, with slow convergence of the stationary distribution of the Markov chain to the target distribution. On the other hand, for high-variance proposal distributions the acceptance rate can be very low because the proposal samples are likely to belong to regions with low probability density, and again convergence to the target distribution can be slow. In the examples discussed in Section 5, we use isotropic Gaussian proposals $q(\Omega^*|\Omega[k])$, with variance chosen through trial-and-error.

[2] Since the matrix $\mathbb{X}_i$ in (9), and thus the parameter β in (12e), depends on the partition $\mathcal{X}[\Omega]$ of the regressor space $\mathcal{X}$. It is thus important to specify in (13) and in the rest of the paper which partition is used to compute $\mathbb{X}_i$ and β.

Algorithm 1 Metropolis-Hastings MCMC for $p(\Omega|X,Y)$

Input: initial value $\Omega[0]$; proposal distribution $q(\Omega^*|\Omega[k])$; number of iterations M.

```
1. for $k = 0, \dots, M-1$ do
    1.1. draw proposal $\Omega^*$ from $q(\Omega^*|\Omega[k])$;
    1.2. set acceptance probability
         $\mathcal{A}(\Omega^*, \Omega[k]) \leftarrow \min\left\{1, \frac{p(\Omega^*|X,Y)q(\Omega[k]|\Omega^*)}{p(\Omega[k]|X,Y)q(\Omega^*|\Omega[k])}\right\}$
         with ratio $\frac{p(\Omega^*|X,Y)}{p(\Omega[k]|X,Y)}$ in (13);
    1.3. accept proposal $\Omega^*$ with probability $\mathcal{A}(\Omega^*, \Omega[k])$;
    1.4. if the proposal $\Omega^*$ is accepted, set $\Omega[k+1] \leftarrow \Omega^*$;
    1.5. else, set $\Omega[k+1] \leftarrow \Omega[k]$;
2. end for;
3. end.
```

Output: Samples $\{\Omega[k]\}_{k=1}^{M}$.

3.3. Making inference

Once an approximation of the posterior $p(\Theta, \Omega, \sigma_v^{-2}|X,Y)$ is computed as in (11), we can make a prediction of the output $y^\star$ given a new test input $x^\star$. According to Bayesian estimation, we look for the distribution $p(y^\star|x^\star, X, Y)$ of $y^\star$ given the input $x^\star$ and the training data X, Y.

The distribution of interest $p(y^\star|x^\star, X, Y)$ is written as

$$p(y^\star|x^\star, X, Y) = \int p(y^\star|x^\star, \Omega, X, Y)p(\Omega|X,Y)d\Omega$$

and then approximated using the empirical mass distribution (10) by

$$p(y^\star|x^\star, X, Y) \approx \frac{1}{M}\sum_{k=1}^{M} p(y^\star|x^\star, \Omega[k], X, Y). \tag{14}$$

Thus, only the conditional distribution $p(y^\star|x^\star, \Omega[k], X, Y)$ is needed in (14). Its expression is provided by the following proposition.

Proposition 3. *Consider the regressor-space partition $\mathcal{X}[\Omega[k]]$ and let $i^\star[k]$ be the index of the polyhedron where $x^\star$ belongs to.*[3] *The output distribution $p(y^\star|x^\star, \Omega[k], X, Y)$ is the multivariate Student distribution*

$$\begin{aligned} &p(y^\star|x^\star, \Omega[k], X, Y) = St(y^\star; \mu_{i^\star}'[k]x^\star, V_{i^\star}[k], 2\alpha) \\ &\propto \left(1 + \frac{1}{2\alpha}(y^\star - \mu_{i^\star}'[k]x^\star)'V_{i^\star}^{-1}[k](y^\star - \mu_{i^\star}'[k]x^\star)\right)^{-\frac{ny+2\alpha}{2}} \end{aligned} \tag{15}$$

with mean $\mu_{i^\star}'[k]x^\star$, diagonal covariance matrix $\frac{\alpha}{\alpha-1}V_{i^\star}[k]$ and degrees of freedom 2α, where

$$V_{i^\star}[k] = \frac{\beta[k]}{\alpha}(x^{\star'}(\mathbb{X}_{i^\star}[k]\mathbb{X}_{i^\star}'[k] + \lambda^{-2}I_{n_\theta})^{-1}x^\star + 1)I_{n_y} \tag{16}$$

and $\mu_{i^\star}[k]$ is defined analogously to (12c) based on the partition $\mathcal{X}[\Omega[k]]$, i.e.,

$$\mu_{i^\star}[k] = \left(\mathbb{X}_{i^\star}[k]\mathbb{X}_{i^\star}'[k] + \lambda^{-2}I_{n_\theta}\right)^{-1}\mathbb{X}_{i^\star}[k]\mathbb{Y}_{i^\star}'[k].$$

See Appendix A.3 for a proof of Proposition 3.

The probability distribution $p(y^\star|x^\star, X, Y)$ in Eq. (14) is thus a mixture of Student distributions. The conditional expected value and covariance matrix of $y^\star$ can be derived using standard results for mixtures of distributions, and they are given by:

$$\begin{aligned} \mathbb{E}\left[y^\star|x^\star, X, Y\right] =& \frac{1}{M}\sum_{k=1}^{M}\mu_{i^\star[k]}'[k]x^\star, \\ Cov\left(y^\star|x^\star, X, Y\right) =& \frac{1}{M}\sum_{k=1}^{M}\frac{\alpha}{\alpha-1}V_{i^\star[k]}[k] \\ &+ \frac{1}{M}\sum_{k=1}^{M}\Bigg((\mu_{i^\star[k]}'[k]x^\star - \frac{1}{M}\sum_{h=1}^{M}\mu_{i^\star[k]}'[h]x^\star) \\ &(\mu_{i^\star[k]}'[k]x^\star - \frac{1}{M}\sum_{h=1}^{M}\mu_{i^\star[k]}'[h]x^\star)'\Bigg). \end{aligned}$$

3.4. Maximum-a-posteriori estimate

In case one is not interested in the posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X,Y)$ but only in seeking for the *maximum-a-posteriori* (MAP) estimate of the parameters $\Theta, \Omega, \sigma_v^{-2}$, the samples generated by Algorithm 1 can be used to approximate the MAP estimate as

$$\begin{aligned} \hat{\Theta}, \hat{\Omega}, \hat{\sigma}_v^{-2} &= \underset{\Theta,\Omega,\sigma_v^{-2}}{\operatorname{argmax}}\, p(\Omega|X,Y)p(\Theta, \sigma_v^{-2}|\Omega, X, Y) \\ &\approx \arg\max_{\{\Omega[k]\}_{k=1}^{M}} p(\Omega[k]|X,Y)\max_{\Theta,\sigma_v^{-2}} p(\Theta, \sigma_v^{-2}|\Omega[k], X, Y). \end{aligned} \tag{18}$$

The values of the parameters $\hat{\Theta}, \hat{\Omega}, \hat{\sigma}_v^{-2}$ solving problem (18) are provided in the following proposition.

Proposition 4. *The MAP estimate for the parameter Ω is given by:*

$$\hat{\Omega} = \underset{\{\Omega[k]\}_{k=1}^{M}}{\operatorname{argmax}}\, \beta[k]^{1-\frac{sn_yn_\theta}{2}-\alpha}p(\Omega[k]). \tag{19}$$

Furthermore, let $\hat{\mu}_i$ (resp. $\hat{\beta}$) be the parameter μ_i (resp. β) in (12c) (resp. in (12e)) associated to the partition $\mathcal{X}[\hat{\Omega}]$ generated by the parameter $\hat{\Omega}$ in (19). Then, the MAP estimate for the parameters θ_i and σ_v^{-2} is given by:

$$\hat{\theta}_i = \hat{\mu}_i, \quad \hat{\sigma}_v^{-2} = \frac{\alpha + \frac{sn_yn_\theta}{2} - 1}{\hat{\beta}}. \tag{20}$$

See Appendix A.4 for a proof of Proposition 4.

The MAP estimate provided in Proposition 4 is based on the outcome $\{\Omega[k]\}_{k=1}^{M}$ of Algorithm 1. This strategy is not efficient, as some samples are generated by exploring regions which might not be around the maximizer of $p(\Theta, \Omega, \sigma_v^{-2}|X,Y)$. Based on the expression of the MAP parameter estimate $\hat{\Omega}$ in (19), a more efficient approach is to generate samples $\Omega[k]$ simulating a Markov chain with invariant probability distribution proportional to

$$\max_{\Theta,\sigma_v^{-2}} p(\Theta, \Omega, \sigma_v^{-2}|X,Y) = \beta^{1-\frac{sn_yn_\theta}{2}-\alpha}p(\Omega). \tag{21}$$

The intuitive idea of generating samples $\Omega[k]$ using (21) instead of the marginal posterior $p(\Omega|X,Y)$ is to approximate, instead of the posterior $p(\Theta, \Omega, \sigma_v^{-2}|X,Y)$, only the manifold $\max_{\Theta,\sigma_v^{-2}} p(\Theta, \Omega, \sigma_v^{-2}|X,Y)$. Furthermore, (21) can be also embedded into a *simulated annealing* strategy to simulate a time-heterogeneous Markov chain with invariant probability distribution at iteration k proportional to

$$\left(\max_{\Theta,\sigma_v^{-2}} p(\Theta, \Omega, \sigma_v^{-2}|X,Y)\right)^{\frac{1}{h[k]}} = \left(\beta^{1-\frac{sn_yn_\theta}{2}-\alpha}p(\Omega)\right)^{\frac{1}{h[k]}}, \tag{22}$$

where $h[k]$ is a decreasing *cooling schedule* with $\lim_{k\to\infty} h[k] = 0$. Simulated annealing is a well known strategy to compute

[3] The dependence of $i^\star[k]$ on k will be omitted to simplify notation.

the maximum of probability distributions and it is motivated by the fact that, as $h[k] \to 0$, all the mass of $(\max_{\Theta,\sigma_v^{-2}} p(\Theta, \Omega, \sigma_v^{-2}|X, Y))^{\frac{1}{h[k]}}$ is concentrated on the parameters Ω maximizing $\max_{\Theta,\sigma_v^{-2}} p(\Theta, \Omega, \sigma_v^{-2}|X, Y)$.

Summarizing, the computation of a maximum-a-posteriori estimate with simulated annealing can be carried out by considering the following acceptance probability $\mathcal{A}(\Omega^*, \Omega[k])$ at Step 1.2 of Algorithm 1:

$$\mathcal{A}(\Omega^*, \Omega[k]) \leftarrow \min\left\{1, \frac{\left((\beta^*)^{1-\frac{sn_yn_\theta}{2}-\alpha} p(\Omega^*)\right)^{\frac{1}{h[k]}} q(\Omega[k]|\Omega^*)}{\left((\beta[k])^{1-\frac{sn_yn_\theta}{2}-\alpha} p(\Omega[k])\right)^{\frac{1}{h[k]}} q(\Omega^*|\Omega[k])}\right\}.$$

4. Rao-Blackwellised particle filters for iterative learning

This section discusses the iterative computation of the posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X, Y)$ through an incremental learning algorithm. By denoting with $X_{1:t}$ (resp. $Y_{1:t}$) the set containing regressors (resp. outputs) from index 1 to index t, the goal is to recursively update $p(\Theta, \Omega, \sigma_v^{-2}|X_{1:t-1}, Y_{1:t-1})$ in order to obtain the posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X_{1:t}, Y_{1:t})$.

A Rao-Blackwellised version of particle filters is proposed. In order to allow for an implementation based on particle filters, the parameter Ω defining the partition of the input space $\mathcal{X}$ is not assumed to be a constant, but it is allowed to vary from index $t-1$ to t. To this end, the variable Ω_t at time t is assumed to be conditionally independent of past $X_{1:t-1}, Y_{1:t-1}, \Omega_{1:t-2}$ given Ω_{t-1}, *i.e.*,

$$p(\Omega_t|\Omega_{1:t-1}, X_{1:t-1}, Y_{1:t-1}) = p(\Omega_t|\Omega_{t-1}). \tag{23}$$

The variation of the parameter Ω from time $t-1$ to t is modelled by the stochastic rule

$$\Omega_t = \Omega_{t-1} + E_\Omega, \tag{24}$$

where $E_\Omega \in \mathbb{R}^{n_\omega,s}$ is a Gaussian random matrix with zero mean and covariance $\sigma_\Omega^2 I_{n_\omega s}$. The matrix E_Ω acts as a fictitious process noise on the variable Ω_t. The parameter σ_Ω^2 should be tuned to trade off between *exploration* over the domain of the parameter Ω and *exploitation*. Indeed, if Ω is treated as a constant parameter (*i.e.*, $\Omega_t = \Omega_{t-1}$, or equivalently $\sigma_\Omega^2 = 0$) then there is no exploration in the particle filter algorithm and the variable $\Omega_t[k]$ will be equal to the initial guess $\Omega_0[k]$ made at the first iteration of the particle filter algorithm. On the other hand, large values of the variance σ_Ω^2 may move particles in regions with low likelihood.

The goal is to recursively compute the joint and the marginal posterior distributions $p(\Theta, \sigma_v^{-2}, \Omega_{1:t}|X_{1:t}, Y_{1:t})$ and $p(\Theta, \sigma_v^{-2}, \Omega_t|X_{1:t}, Y_{1:t})$, with $\Omega_{1:t}$ denoting the sequence of the parameters Ω from index 1 to index t.

4.1. Rao-Blackwellised approach

Similarly to the batch Rao-Blackwellised MCMC algorithm discussed in Section 3.1, the posterior distribution is first factorized as

$$\begin{aligned} &p(\Theta, \sigma_v^{-2}, \Omega_{1:t}|X_{1:t}, Y_{1:t}) \\ =&p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t})p(\Omega_{1:t}|X_{1:t}, Y_{1:t}). \end{aligned} \tag{25}$$

The conditional distribution $p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t})$ is updated (recursively) analytically, while only the marginal posterior $p(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ is approximated through particle filters.

Let $s_{t|t}$ be the active mode, at the index t, defined based on the regressor-space partition $\mathcal{X}[\Omega_t]$, *i.e.*, $s_{t|t} = i \Leftrightarrow x_t \in \mathcal{X}_i[\Omega_t]$. The matrices $\mathbb{Y}_{i,t}$ and $\mathbb{X}_{i,t}$ at index t can be redefined as: y_t' is a row of $\mathbb{Y}_{i,t} \Leftrightarrow s_{t|t} = i$ and $\left[\begin{smallmatrix}1\\ x_t\end{smallmatrix}\right]$ = kth column of $\mathbb{X}_{i,t}$ $\Leftrightarrow$ y_t' = kth row of $\mathbb{Y}_{i,t}$. These matrices can be constructed recursively as follows

$$\mathbb{Y}_{i,t} = \begin{cases} \begin{bmatrix} \mathbb{Y}_{i,t-1} \\ y_t' \end{bmatrix} & \text{if } s_{t|t} = i, \\ \mathbb{Y}_{i,t-1} & \text{otherwise.} \end{cases} \tag{26a}$$

$$\mathbb{X}_{i,t} = \begin{cases} \begin{bmatrix} \mathbb{X}_{i,t-1} & \begin{smallmatrix}1\\ x_t\end{smallmatrix} \end{bmatrix} & \text{if } s_{t|t} = i, \\ \mathbb{X}_{i,t-1} & \text{otherwise.} \end{cases} \tag{26b}$$

Note that the matrices $\mathbb{Y}_{i,t}$ and $\mathbb{X}_{i,t}$ depend on the regressor-space partitions $\mathcal{X}[\Omega_1], \ldots, \mathcal{X}[\Omega_t]$. If necessary, this dependence will be made explicit as $\mathbb{Y}_{i,t}[\Omega_{1:t}]$ and $\mathbb{X}_{i,t}[\Omega_{1:t}]$.

The conditional distribution $p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t})$ can be computed straightforwardly using the same derivations in Proposition 1, taking into account the index-dependence of the parameter Ω. More specifically,

$$\begin{aligned} &p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t}) \\ =&\underbrace{\Gamma(\sigma_v^{-2}; \alpha_t, \beta_t)}_{p(\sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t})}\underbrace{\prod_{i=1}^{s}\prod_{j=1}^{n_y}\mathcal{N}(\theta_i^{(j)}; \mu_{i,t}^{(j)}, \sigma_v^2 F_{i,t})}_{p(\Theta|\sigma_v^{-2}, \Omega_{1:t}, X_{1:t}, Y_{1:t})}, \end{aligned} \tag{27}$$

with $F_{i,t}$, $\mu_{i,t}$, α_t and β_t defined similarly to (12). Specifically,

$$F_{i,t}[\Omega_{1:t}] = \left(\mathbb{X}_{i,t}\mathbb{X}_{i,t}' + \lambda^{-2}I_{n_\theta}\right)^{-1}, \tag{28a}$$

$$\mu_{i,t}[\Omega_{1:t}] = \left(\mathbb{X}_{i,t}\mathbb{X}_{i,t}' + \lambda^{-2}I_{n_\theta}\right)^{-1}\mathbb{X}_{i,t}\mathbb{Y}_{i,t}, \tag{28b}$$

$$\alpha_t = \alpha_0 + \frac{n_y t}{2}, \tag{28c}$$

$$\beta_t[\Omega_{1:t}] = \beta_0 + \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}\left(\mathbb{Y}_{i,t}^{(j)'}\mathbb{Y}_{i,t}^{(j)} - \mathbb{Y}_{i,t}^{(j)'}\mathbb{X}_{i,t}'\mu_{i,t}^{(j)}\right), \tag{28d}$$

where the dependence on the parameters $\Omega_{1:t}$ is made explicit in the definitions above.

The following proposition provides the formulas to recursively update the parameters $F_{i,t}$, $\mu_{i,t}$ and β_t that characterize the conditional distribution $p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t})$ in (27).

Proposition 5. *The parameters $F_{i,t}$ (and $F_{i,t}^{-1}$), $\mu_{i,t}$ and β_t defined in (28a), (28b) and (28d) can be recursively updated as follows:*

$$F_{i,t}^{-1} = F_{i,t-1}^{-1} + \left[\begin{smallmatrix}1\\ x_t\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & x_t\end{smallmatrix}\right]\mathbb{I}(s_{t|t} = i), \tag{29a}$$

$$F_{i,t} = F_{i,t-1} - \frac{F_{i,t-1}\left[\begin{smallmatrix}1\\ x_t\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & x_t\end{smallmatrix}\right]F_{i,t-1}}{1 + \left[\begin{smallmatrix}1 & x_t\end{smallmatrix}\right]F_{i,t-1}\left[\begin{smallmatrix}1\\ x_t\end{smallmatrix}\right]}\mathbb{I}(s_{t|t} = i), \tag{29b}$$

$$\mu_{i,t} = \mu_{i,t-1} + F_{i,t}\left[\begin{smallmatrix}1\\ x_t\end{smallmatrix}\right](y_t' - \left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{i,t-1})\mathbb{I}(s_{t|t} = i) \tag{29c}$$

$$\begin{aligned} \beta_t =& \beta_{t-1} + \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}(y_t^{(j)}y_t^{(j)} - y_t^{(j)}\left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{i,t}^{(j)})\mathbb{I}(s_{t|t} = i) \\ &- \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}\mu_{i,t-1}^{(j)'}\left(I_{n_\theta} - \left[\begin{smallmatrix}1\\ x_t\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & x_t\end{smallmatrix}\right]F_{i,t}\right)\left[\begin{smallmatrix}1\\ x_t\end{smallmatrix}\right] \\ &\times (y_t^{(j)} - \left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{i,t-1}^{(j)})\mathbb{I}(s_{t|t} = i). \end{aligned} \tag{29d}$$

See Appendix A.5 for a proof of Proposition 5.

4.2. Update of $p(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ through particle filters

We discuss now the recursive update of the marginal posterior $p(\Omega_{1:t}|X_{1:t}, Y_{1:t})$. According to particle filter algorithms, the distribution $p(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ is approximated by the empirical point-mass distribution

$$p(\Omega_{1:t}|X_{1:t}, Y_{1:t}) \approx \sum_{k=1}^{N_p} w_t[k]\delta_{\Omega_{1:t}[k]}(\Omega_{1:t}), \quad \sum_{k=1}^{N_p} w_t[k] = 1, \tag{30}$$

Algorithm 2 Updating particles' position $\{\Omega_t[k]\}_{k=1}^{N_\mathrm{p}}$ and weights $\{w_t[k]\}_{k=1}^{N_\mathrm{p}}$

Input: previous particles' position $\{\Omega_{t-1}[k]\}_{k=1}^{N_\mathrm{p}}$ and weights $\{w_{t-1}[k]\}_{k=1}^{N_\mathrm{p}}$; proposal distribution $q(\Omega_t|\Omega_{t-1})$; current output y_t and input x_t.

```
1. for $k = 1, \ldots, N_\mathrm{p}$ do
    1.1. resample $\tilde{\Omega}_{t-1}[k]$ from probability distribution
         $\sum_{k=1}^{N_\mathrm{p}} w_{t-1}[k]\delta_{\Omega_{t-1}[k]}(\tilde{\Omega}_{t-1})$;
    1.2. set $\tilde{w}_{t-1}[k] = \frac{1}{N_\mathrm{p}}$;
2. end for;
3. for $k = 1, \ldots, N_\mathrm{p}$ do
    3.1. set $w_{t-1}[k] \leftarrow \tilde{w}_{t-1}[k],\ \ \Omega_{t-1}[k] \leftarrow \tilde{\Omega}_{t-1}[k]$;
    3.2. generate sample $\Omega_t[k]$ from $q(\Omega_t|\Omega_{t-1}[k])$;
    3.3. set weights
         $\tilde{w}_t[k]\leftarrow p(y_t|\Omega_{1:t}[k],X_{1:t-1},x_t,Y_{1:t-1})\frac{p(\Omega_t|\Omega_{t-1}[k])}{q(\Omega_t|\Omega_{t-1}[k])}w_{t-1}[k]$
         with $p(y_t|\Omega_{1:t}[k],X_{1:t-1},x_t,Y_{1:t-1})$ in (31);
4. end for;
5. normalize weights $w_t[k] \leftarrow \frac{\tilde{w}_t[k]}{\sum_{j=1}^{N_\mathrm{p}} \tilde{w}_t[j]},\ \ k = 1, \ldots, N_\mathrm{p}$;
6. end.
```

Output: current particles' position $\{\Omega_t[k]\}_{k=1}^{N_\mathrm{p}}$ and weights $\{w_t[k]\}_{k=1}^{N_\mathrm{p}}$.

where N_p is the number of particles, $\Omega_{1:t}[k]$ is the trajectory of the kth particle from index 1 to index t, and $w_t[k]$ is a non-negative weight associated to the particle.

The marginal posterior distribution $p(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ is factorized (up to the scaling factor $\frac{1}{p(y_t|Y_{1:t-1})}$) as

$$p(\Omega_{1:t}|X_{1:t}, Y_{1:t}) \propto$$
$$p(y_t|\Omega_{1:t}, X_{1:t}, Y_{1:t-1})p(\Omega_t|\Omega_{t-1})p(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1}),$$

and $p(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1})$ is approximated as in (30) based on the trajectory of the particles up to $t-1$, *i.e.*,

$$p(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1}) \approx \sum_{k=1}^{N_\mathrm{p}} w_{t-1}[k]\delta_{\Omega_{1:t-1}[k]}(\Omega_{1:t-1}).$$

The particles' weights $w_t[k]$ are computed recursively based on the weights $w_{t-1}[k]$ using a standard particle filter approach outlined in Algorithm 2. At Steps 1.1–1.2, equally-weighted particles are generated by resampling the variable $\Omega_{t-1}[k]$, $k = 1, \ldots, N_\mathrm{p}$, from the previous approximation of the marginal distribution $p(\Omega_{t-1}|X_{1:t-1}, Y_{1:t-1})$. A new sample $\Omega_t[k]$ is then generated from a proposal distribution $q(\Omega_t|\Omega_{t-1}[k])$ (Step 3.2) and the particles' weights $w_t[k]$ are updated and finally normalized (Steps 3.3 and 5). The generated samples $\{\Omega_t[k]\}_{k=1}^{N_\mathrm{p}}$ and normalized weights $\{w_t[k]\}_{k=1}^{N_\mathrm{p}}$ are then used to approximate the distribution $p(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ as in (30). As a common practice in particle filtering, the proposal $q(\Omega_t|\Omega_{t-1}[k])$ is chosen to be $p(\Omega_t|\Omega_{t-1}[k])$. Thus, based on the assumptions on the evolution of the parameter Ω_t in (24), the new matrix $\Omega_t[k]$ is generated by a Gaussian distribution with mean $\Omega_{t-1}[k]$ and covariance $\sigma_\Omega^2 I_{n_\omega s}$.

Implementing Algorithm 2 requires computing the marginal likelihood $p(y_t|\Omega_{1:t}[k], X_{1:t-1}, x_t, Y_{1:t-1})$ at Step 3.3, whose expression is provided (up to a proportionality constant) in the following proposition.

Proposition 6. *The marginal likelihood $p(y_t|\Omega_{1:t}[k], X_{1:t-1}, x_t, Y_{1:t-1})$ is proportional to*

$$\frac{(\beta_{t-1}[k])^{\alpha_t-1}}{(\beta_t[k])^{\alpha_t}} \left|F_{s_{t|t}[k],t-1}[k]\right|^{-\frac{n_y}{2}} \left|F^{-1}_{s_{t|t}[k],t}[k]\right|^{-\frac{n_y}{2}}, \tag{31}$$

where $s_{t|t}[k]$ is the active mode at index t associated to the partition $\mathcal{X}[\Omega_t[k]]$, i.e., *$s_{t|t}[k] = i \Leftrightarrow x_t \in \mathcal{X}_i[\Omega_t[k]]$.*

In (31), $F_{i,t}[k]$ and $\beta_t[k]$ are used as a short notation for $F_{i,t}[\Omega_{1:t}[k]]$ and $\beta_t[\Omega_{1:t}[k]]$ in (28).

A proof of the proposition is provided in Appendix A.6.

Once the particles' weights $\{w_t[k]\}_{k=1}^{N_\mathrm{p}}$ are computed through Algorithm 2, the joint posterior distribution $p(\Theta, \sigma_v^{-2}, \Omega_{1:t}|X_{1:t}, Y_{1:t})$ is finally obtained from (25), (27) and (30), *i.e.*,

$$\begin{aligned}
&p(\Theta, \sigma_v^{-2}, \Omega_{1:t}|X_{1:t}, Y_{1:t}) \\
&= \sum_{k=1}^{N_\mathrm{p}} w_t[k]p(\Theta, \sigma_v^{-2}|\Omega_{1:t}[k], X_{1:t}, Y_{1:t})\delta_{\Omega_{1:t}[k]}(\Omega_{1:t}) \\
&= \sum_{k=1}^{N_\mathrm{p}} w_t[k]\Gamma(\sigma_v^{-2}; \alpha_t, \beta_t[k]) \\
&\quad \times \prod_{i=1}^{s}\prod_{j=1}^{n_y} \mathcal{N}(\theta_i^{(j)}; \mu_{i,t}^{(j)}[k], \sigma_v^2 F_{i,t}[k])\delta_{\Omega_{1:t}[k]}(\Omega_{1:t}),
\end{aligned} \tag{32}$$

where $\mu_{i,t}[k]$ is a short for $\mu_{i,t}[\Omega_{1:t}[k]]$ in (28b).

The marginal distribution $p(\Theta, \sigma_v^{-2}, \Omega_t|X_{1:t}, Y_{1:t})$ is derived from $p(\Theta, \sigma_v^{-2}, \Omega_{1:t}|X_{1:t}, Y_{1:t})$ and approximated as

$$\begin{aligned}
&p(\Theta, \sigma_v^{-2}, \Omega_t|X_{1:t}, Y_{1:t}) \\
&\approx \sum_{k=1}^{N_\mathrm{p}} w_t[k]p(\Theta, \sigma_v^{-2}|\Omega_{1:t}[k], X_{1:t}, Y_{1:t})\delta_{\Omega_t[k]}(\Omega_t) \\
&= \sum_{k=1}^{N_\mathrm{p}} w_t[k]\Gamma(\sigma_v^{-2}; \alpha_t, \beta_t[k]) \\
&\quad \times \prod_{i=1}^{s}\prod_{j=1}^{n_y} \mathcal{N}(\theta_i^{(j)}; \mu_{i,t}^{(j)}[k], \sigma_v^2 F_{i,t}[k])\delta_{\Omega_t[k]}(\Omega_t).
\end{aligned} \tag{33}$$

4.3. Making inference

The distribution $p(y^\star|x^\star, X_{1:t}, Y_{1:t})$ of a predictor $y^\star$ given a new test input $x^\star$ can be computed using the marginal distribution $p(\Theta, \sigma_v^{-2}, \Omega_t|X_{1:t}, Y_{1:t})$ in (33). Specifically, let us consider the partition $\mathcal{X}[\Omega_t[k]]$ and let $i_t^\star[k]$ be the index of the region where $x^\star$ belong to, *i.e.*,

$$i_t^\star[k] : x^\star \in \mathcal{X}_{i_t^\star[k]}[\Omega_t[k]].$$

The dependence of $i_t^\star[k]$ on t and k will be omitted in the following. Based on the same arguments in Section 3.3, the distribution $p(y^\star|x^\star, X_{1:t}, Y_{1:t})$ is approximated by

$$p(y^\star|x^\star, X_{1:t}, Y_{1:t}) \approx \frac{1}{N_\mathrm{p}} \sum_{k=1}^{N_\mathrm{p}} p(y^\star|x^\star, \Omega_t[k], X_{1:t}, Y_{1:t}),$$

where $p(y^\star|x^\star, \Omega_t[k], X_{1:t}, Y_{1:t})$ is the multivariate t-Student distribution

$$\begin{aligned}
&p(y^\star|x^\star, \Omega_t[k], X_{1:t}, Y_{1:t}) = St(y^\star; \mu'_{i^\star,t}[k]x^\star, V_{i^\star,t}[k], 2\alpha_t) \\
&\propto \left(1 + \frac{1}{2\alpha_t}(y^\star - \mu'_{i^\star,t}[k]x^\star)'V^{-1}_{i^\star,t}[k](y^\star - \mu'_{i^\star,t}[k]x^\star)\right)^{-\frac{n_y+2\alpha_t}{2}},
\end{aligned}$$

with $V_{i^\star,t}[k]$ defined as in (16) based on the polyhedral partitions $\mathcal{X}[\Omega_1[k]], \ldots, \mathcal{X}[\Omega_t[k]]$, *i.e.*,

$$V_{i^\star,t}[k] = \frac{\beta_t[k]}{\alpha_t}(x^{\star'}(\mathbb{X}_{i^\star,t}[k]\mathbb{X}'_{i^\star,t}[k] + \lambda^{-2}I_{n_\theta})^{-1}x^\star + 1)I_{n_y}.$$

4.4. Maximum-a-posteriori estimate

In case the final objective is only to seek for the maximum-a-posteriori estimate

$$\hat{\Theta}_t, \hat{\sigma}_{v,t}^{-2}, \hat{\Omega}_{1:t} = \underset{\Theta,\sigma_v^{-2},\Omega_{1:t}}{\operatorname{argmax}}\ p(\Theta, \sigma_v^{-2}, \Omega_{1:t}|X_{1:t}, Y_{1:t}), \tag{34}$$

an approach similar to the one discussed in Section 3.4 can be used. Specifically, the value of the MAP parameter estimate $\hat{\Theta}_t, \hat{\sigma}_{v,t}^{-2}, \hat{\Omega}_{1:t}$ solving problem (34) is provided in the following proposition.

Proposition 7. *The MAP estimate for the parameters $\Omega_{1:t}, \theta_{i,t}, \sigma_{v,t}^{-2}$ is given by:*

$$\hat{\Omega}_{1:t} = \underset{\Omega_{1:t}}{\operatorname{argmax}}\, p(\Omega_{1:t}|X_{1:t}, Y_{1:t})\beta_t^{1-\frac{sn_yn_\theta}{2}} \prod_{i=1}^{s}|F_{i,t}^{-1}|^{\frac{n_y}{2}}, \tag{35a}$$

$$\hat{\theta}_{i,t} = \mu_{i,t}[\hat{\Omega}_{1:t}], \quad \hat{\sigma}_{v,t}^{-2} = \frac{\alpha + \frac{sn_yn_\theta}{2} - 1}{\beta_t[\hat{\Omega}_{1:t}]}, \tag{35b}$$

where $\mu_{i,t}[\hat{\Omega}_{1:t}]$ (resp. $\beta_t[\hat{\Omega}_{1:t}]$) is defined in (28b) (resp. in (28d)) for $\Omega_{1:t} = \hat{\Omega}_{1:t}$.

See Appendix A.7 for a derivation of the results in Proposition 7.

The following proposition provides a recursive formula to iteratively update the objective function in (35a).

Proposition 8. *To compact the notation, let us denote the objective function in (35a) with $\tilde{p}(\Omega_{1:t}|X_{1:t}, Y_{1:t})$,* i.e.,

$$\tilde{p}(\Omega_{1:t}|X_{1:t}, Y_{1:t}) = p(\Omega_{1:t}|X_{1:t}, Y_{1:t})\beta_t^{1-\frac{sn_yn_\theta}{2}} \prod_{i=1}^{s}|F_{i,t}^{-1}|^{\frac{n_y}{2}}$$

The following formula provides a recursive formula to compute $\tilde{p}(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ (up to a normalization constant) starting from $\tilde{p}(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1})$:

$$\begin{aligned}&\tilde{p}(\Omega_{1:t}|X_{1:t}, Y_{1:t})\\ &\propto \frac{(\beta_t)^{1-\frac{sn_yn_\theta}{2}-\alpha_t}}{(\beta_{t-1})^{1-\frac{sn_yn_\theta}{2}-\alpha_{t-1}}} p(\Omega_t|\Omega_{t-1})\tilde{p}(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1}).\end{aligned} \tag{36}$$

See Appendix A.8 for a proof of Proposition 8.

Using the recursive formula in (36), the MAP estimation problem (35a) is solved by approximating its objective function $\tilde{p}(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ via particle filters:

$$\hat{\Omega}_{1:t} \approx \arg \max_{\{\Omega_{1:t}[k]\}_{k=1}^{N_\mathrm{p}}} \tilde{p}(\Omega_{1:t}[k]|X_{1:t}, Y_{1:t}). \tag{37}$$

The particles' trajectory $\{\Omega_{1:t}[k]\}_{k=1}^{N_\mathrm{p}}$ and the associated weights $\{w_t[k]\}_{k=1}^{N_\mathrm{p}}$ approximating $\tilde{p}(\Omega_{1:t}[k]|X_{1:t}, Y_{1:t})$ are computed through Algorithm 2, by simply replacing the (unnormalized) weight update at Step 3.3 with

$$\tilde{w}_t[k] \leftarrow \frac{(\beta_t[k])^{1-\frac{sn_yn_\theta}{2}-\alpha_t}}{(\beta_{t-1}[k])^{1-\frac{sn_yn_\theta}{2}-\alpha_{t-1}}} \frac{p(\Omega_t|\Omega_{t-1}[k])}{q(\Omega_t|\Omega_{t-1}[k])} w_{t-1}[k].$$

Then, an approximation of the MAP estimate $\hat{\Omega}_{1:t}$ can be finally computed as

$$\hat{\Omega}_{1:t} \approx \arg \max_{\{\Omega_{1:t}[k]\}_{k=1}^{N_\mathrm{p}}} \tilde{p}(\Omega_{1:t}[k]|X_{1:t}, Y_{1:t}). \tag{38}$$

Once the polyhedral partition $\mathcal{X}[\hat{\Omega}_t]$ is computed, the MAP estimates $\hat{\theta}_{i,t}$ and $\hat{\sigma}_{v,t}^{-2}$ are from (35b) and using the recursive formulas (28).

5. Examples and applications

The algorithms presented in the previous sections are tested via a numerical example using synthetic data and through a benchmark case study. The tests are run on an i7 2.40-GHz Intel core processor in MATLAB R2016b. MATLAB codes of the algorithms can be found at http://dariopiga.com/Software/PWABay.rar.

In both examples the hyper-parameters $\sigma_\omega^2, \alpha_0, \beta_0, \lambda^2$ defining the priors over the PWA model parameters $\Omega, \Theta, \sigma_v^{-2}$ (see Section 2.3) are set to $\sigma_\omega^2 = 100$, $\alpha_0 = 1$, $\beta_0 = 0.001$, $\lambda^2 = 1000$. These hyper-parameters correspond to broad (large variance) "uninformative" prior distributions.

The performance of the estimated models is assessed on a test dataset (separated from the training set) in terms of the *Best Fit Rate* (BFR) defined as

$$\mathrm{BFR} = 1 - \sqrt{\frac{\sum_{t=1}^{T_v}(y_t^\star - \hat{y}_t^\star)'(y_t^\star - \hat{y}_t^\star)}{\sum_{t=1}^{T_v}(y_t^\star - \bar{y}^\star)'(y_t^\star - \bar{y}^\star)}},$$

where T_v is the length of the test sequence, $\hat{y}_t^\star$ is the estimated expected value of the output, $y_t^\star$ and $\bar{y}^\star$ are the true output and its sample mean, respectively.

5.1. Numerical example

Data description

Data is generated by a discontinuous multi-input multi-output PWA function as in (1), with $s = 3$ modes, $y_t \in \mathbb{R}^4$ and $x_t \in \mathbb{R}^{10}$. The matrices $\theta_1, \theta_2, \theta_3 \in \mathbb{R}^{11,4}$ are randomly generated and the regions $\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3$ partitioning the regressor space are defined as in (2), with equation in Box I.

Two disjoint sets of length $T = 20,000$ and $T_v = 1,000$ are generated and used to train the model and assess its performance. The output noise $v_t \in \mathbb{R}^4$ is generated by a zero-mean Gaussian white noise process with covariance matrix $\sigma_v^2 I_4$, with $\sigma_v^2 = 4$. This corresponds to the *signal-to-noise ratio* SNR $= 10\log_{10}\frac{\sum_{t=1}^{T} y_t' y_t}{\sum_{t=1}^{T} v_t' v_t} = 18.5$ dB. To better assess the quality of the estimated model, test data is not corrupted by noise.

Batch learning

First, a batch Bayesian inference problem is addressed. The conditional posterior distribution $p(\Omega|X, Y)$ is approximated by running Algorithm 1 for $M = 5,000$ iterations, with a random initial guess $\Omega[0]$ and an isotropic Gaussian proposal distribution $q(\Omega^*|\Omega[k])$ with mean $\Omega[k]$ and covariance matrix $0.25I_{n_\omega s}$.

Once the posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X, Y)$ is computed, the distribution $p(y_t^\star|x_t^\star, X, Y)$ of the output $y_t^\star$ given a test input $x_t^\star$ is derived as discussed in Section 3.3. Fig. 1 shows the expected value $\hat{y}_t^\star$ of the output, along with the 99%-credible intervals[4] and the true output $y_t^\star$. For the sake of space and for a better visualization, only a subset of test data on the first output is plotted. Similar results are obtained for the other three outputs. The achieved BFR is 93%.

The mean value of the parameters Ω defining the polyhedral partition of the input domain is approximated from the computed marginal posterior $p(\Omega|X, Y)$ and used to estimate the sequence

[4] The credible intervals are computed by approximating the distributions (14)–(15) through numerical sampling.

$$\Omega = [\omega_1 \;\; \omega_2 \;\; \omega_3] = \begin{bmatrix} -0.8 & 0.0 & 0.2 & 3.0 & 0.3 & 0.2 & 0.0 & 1.0 & 5.0 & -0.5 & -0.6 \\ -1.0 & -0.4 & 0.4 & 2.0 & 0.6 & 0.2 & -0.4 & 0.4 & -0.3 & 0.3 & 0.5 \\ -0.5 & 0.2 & 0.7 & 1.5 & 1.0 & -0.2 & 0.7 & -0.2 & 0.2 & 0.5 & 0.8 \end{bmatrix}'$$

Box I.

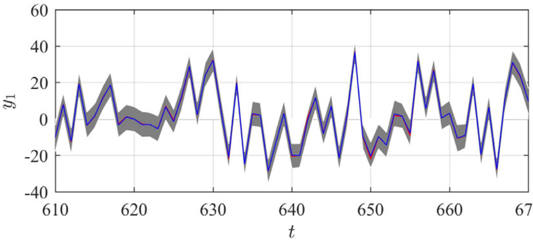


Fig. 1. Batch learning. True output y_t (red); expected value of the output $\hat{y}_t^\star$ (blue); 99%-credible intervals (grey regions). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

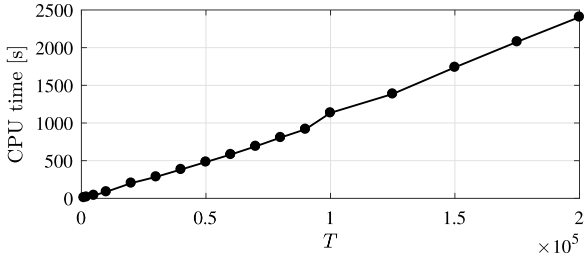


Fig. 2. Batch learning. CPU time required to run Algorithm 1 vs length T of the training dataset.

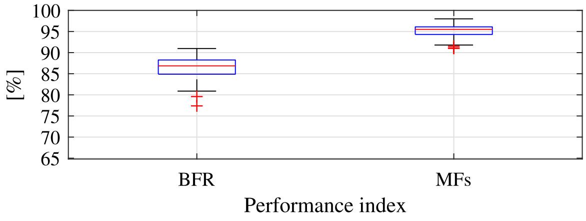


Fig. 3. Batch learning. Monte Carlo analysis: box-plots of achieved BFR and mode-fit index MF$_s$.

of active modes $\{\hat{s}_t^\star\}_{t=1}^{T_v}$. This sequence is compared with the true mode sequence $\{s_t^\star\}_{t=1}^{T_v}$, which is assumed to be provided by an oracle only for validation purposes. The accuracy in reconstructing the hidden mode $s_t^\star$ is measured by the *mode-fit* (MF$_s$) index, defined as $\text{MF}_s = \dfrac{1}{T_v}\displaystyle\sum_{t=1}^{T_v} \mathbb{I}(s_t^\star = \hat{s}_t^\star)$. The resulting MF$_s$ is 98% (namely, 98% of regressor samples $x_t^\star$ are assigned to the "true" local submodel).

Computational complexity analysis

In order to analyse the computational complexity of the proposed learning algorithm, the training phase is performed using training sets of different lengths T. Fig. 2 shows the CPU time required by Algorithm 1 to process the training datasets as a function of T (for a fixed simulation length $M = 5,000$). As expected, the CPU time increases linearly with the length T of the training set. This is due to the fact that the number of operations needed to compute the acceptance probability $\mathcal{A}(\Omega^*, \Omega[k])$ (Algorithm 1, Step 1.2), or equivalently the ratio $\frac{p(\Omega^*|X,Y)}{p(\Omega[k]|X,Y)}$, increases linearly with T. In fact, computing $\frac{p(\Omega^*|X,Y)}{p(\Omega[k]|X,Y)}$ requires to construct $\mathbb{X}_i^\star$, $\mathbb{X}_i[k]$, $\beta^\star$ and $\beta[k]$ (Proposition 2, Eq. (13)). The cost of constructing these parameters increases linearly with T.

Monte Carlo analysis

In order to provide more representative results, a Monte Carlo analysis of 100 runs is performed. At each run, new realizations of the input and noise are generated, and the model parameters Θ and Ω previously considered are randomly perturbed. Furthermore, in order to assess the sensitivity of the algorithm with respect to the hyper-parameters $\sigma_\omega^2, \alpha_0, \beta_0, \lambda^2$ defining the priors on Ω, Θ, σ_v^{-2} (see Section 2.3), $\sigma_\omega^2, \alpha_0, \beta_0, \lambda^2$ are randomly generated at each run from uniform distributions in the intervals [50 150], [1 10], [0.0005 0.0015], and [500 1500], respectively. The width of these intervals is chosen to maintain broad uninformative prior distributions. The proposal $q(\Omega^*|\Omega[k])$ is not tuned, and the same isotropic Gaussian proposal $q(\Omega^*|\Omega[k])$, with diagonal covariance matrix $0.25 I_{n_\omega s}$ is used at each run.

At each Monte Carlo run, the *maximum-a-posteriori* estimate of the model parameters is computed. The box-plots of the achieved BFR and mode-fit index MF$_s$ are reported in Fig. 3, where it can be seen that, except for few outliers, the BFR is between 81% and 91%, with a mode-fit index MF$_s$ between 91% and 98%.

Recursive learning

The recursive learning approach based on particle filters presented in Section 4 is applied next. The samples of the 20, 000-length training set previously used for batch learning are now processed sequentially, thus simulating a scenario where data is gathered and processed in real time. At each index t, the marginal posterior $p(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ is updated by Algorithm 2 using $N_\text{p} = 250$ particles. A proposal distribution $q(\Omega_t|\Omega_t[k-1]) = p(\Omega_t|\Omega_t[k-1])$ is chosen. According to the modelling assumption (24), $p(\Omega_t|\Omega_t[k-1])$ is an isotropic Gaussian distribution centred at $\Omega_t[k-1]$ with diagonal covariance matrix $\sigma_\Omega^2 I_{sn_\omega}$, with $\sigma_\Omega^2 = 0.25$ chosen through trial-and-error.

The average CPU time required by Algorithm 2 to process an input–output pair $\{x_t, y_t\}$ is 18.8 ms. Thus, 20,000 training samples are processed in 354 s.

Inferences on test outputs $y_t^\star$ are made according to the results discussed in Section 4.3, using the "last" marginal distribution $p(\Omega_T|X_{1:T}, Y_{1:T})$ and the corresponding joint posterior $p(\Theta, \sigma_v^{-2}, \Omega_T|X_{1:T}, Y_{1:T})$ in (33). The expected value of the first output $\hat{y}_t^\star$, the 99%-credible intervals and the true output $y_t^\star$ are plotted in Fig. 4. The resulting BFR and *mode-fit* index MF$_s$ are equal to 89% and 96%, respectively.

The obtained results show that the models estimated using the batch and the recursive learning approach achieve similar performance in reconstructing the input-to-output relation. In terms of CPU time required to process a given set of training

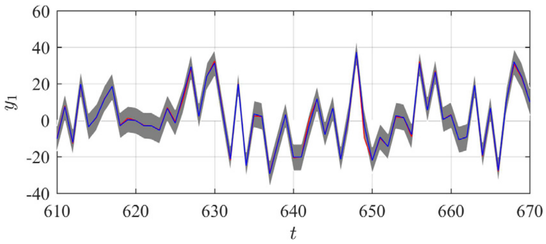


Fig. 4. Iterative learning. True output y_t (red); expected value of the output $\hat{y}_t^{\star}$ (blue); 99%-credible intervals (grey regions). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

data, the batch algorithm is 1.8× faster than the recursive one. However, the recursive approach has the advantages of being suitable for online applications, in which a new data is processed as soon as it becomes available, without the need of storing and reprocessing the entire past data sequence.

5.2. Experimental case study

System description

We consider the case study proposed in Juloski, Heemels, and Ferrari-Trecate (2004) concerning data-driven modelling of the placement process in a pick-and-place machine. The process is characterized by two main operating modes, the *free* and the *impact mode*. In free mode the machine moves the component in an unconstrained environment, while in the impact mode the mounting head is in contact with the board. This process is commonly used as a benchmark to assess the effectiveness of learning algorithms for hybrid dynamical systems (Bemporad, Breschi, Piga, & Boyd, 2018; Bemporad et al., 2005; Juloski et al., 2005; Ohlsson & Ljung, 2013; Pillonetto, 2016).

A data record over an interval of 15 s is gathered at a sampling frequency of 800 Hz. The data record is split into two disjoint subsets: a training set with $T = 8,800$ samples gathered in the first 11 s of the experiment, and a test set with $T_v = 3,200$ samples, gathered in the remaining 4 s. The input u_t is the voltage applied to the motor driving the mounting head, while the output y_t of interest is the vertical position of the mounting head.

A 2-mode PWA dynamical model with regressor $x_t = [1 \;\; y_{t-1} \;\; y_{t-2} \;\; u_{t-1} \;\; u_{t-2}]'$ is used to describe the process behaviour.

Bayesian inference

First, the problem is addressed in a Bayesian setting and the posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X, Y)$ is approximated. Data is processed in batch mode running Algorithm 1 for $M = 5,000$ iterations, using a Gaussian proposal distribution $q(\Omega^{*}|\Omega[k])$ with mean $\Omega[k]$ and diagonal covariance matrix $0.16 I_{n_\omega s}$. For an exhaustive analysis of the proposed methods, training data is also processed iteratively running Algorithm 2 with $N_{\mathrm{p}} = 250$ particles and proposal distribution $q(\Omega_t|\Omega_t[k-1])$ equal to $p(\Omega_t|\Omega_t[k-1])$. The variance σ_Ω^2 of the fictitious process noise E_Ω in (24) is set equal to 0.16.

1000 different parameters $\Theta, \Omega, \sigma_v^{-2}$ are drawn from the computed posterior distribution $p(\Theta, \Omega, \sigma_v^{-2}|X, Y)$ and the output of the corresponding PWA models is simulated. Fig. 5 shows the mean of the output over the 1000 simulations ±3 times the standard deviation.

Table 1
Achieved BFR and CPU time required for training.

Approach	BFR	time
Batch learning	84.9%	20 s
Recursive learning	81.7%	21 s
Clustering-based approach (Ferrari-Trecate et al., 2003)	76.7%	1133 s
Opt.-based approach (Bemporad et al., 2018)	82.4%	0.14 s

Maximum-a-posteriori estimate

The batch and recursive learning algorithms are also used to compute a *maximum-a-posterior* estimate of the model parameters. A *simulated annealing* strategy is implemented (with *cooling schedule* $h[k] = \log(k)$ in (22)) and Algorithms 1 and 2 are modified as discussed in Section 3.4 and 4.4. The other algorithms' settings are the same as the ones described in the previous paragraph.

For comparison, the same PWA regression problem is also solved via the clustering approach[5] in Ferrari-Trecate et al. (2003) and the optimization-based method in Bemporad et al. (2018).

The open-loop predicted outputs of the PWA models estimated using the different learning algorithms are plotted in Fig. 6. The resulting BFRs are reported in Table 1, along with the CPU time required to process the entire dataset for fixed tuning parameters. It can be observed that the approach in Ferrari-Trecate et al. (2003) is more than 53x slower than the batch and the iterative learning algorithms proposed in this paper, while the method in Bemporad et al. (2018) is the fastest one. However, we remark that Bemporad et al. (2018) can only process data in a batch mode, while the particle-filter based algorithm proposed in this paper can also process streams of data in an iterative way.

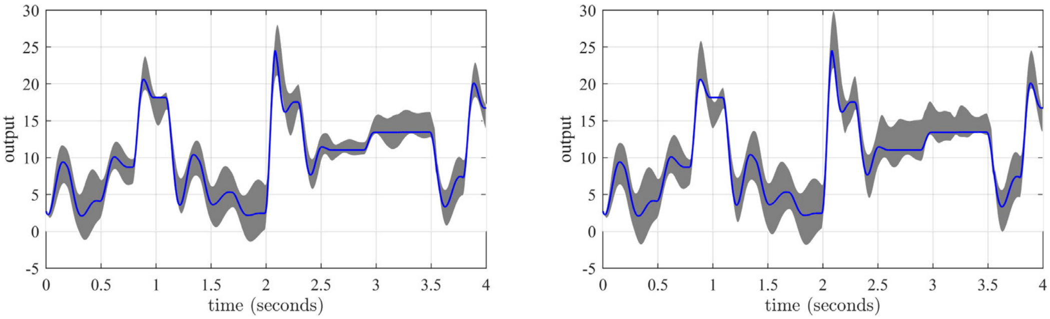
6. Conclusions

This paper has discussed a unified framework for batch and recursive Bayesian inference of PieceWise Affine models using Rao-Blackwellized Monte Carlo sampling. Instead of approximating the joint posterior distribution of the model parameters through *naive* Monte Carlo sampling, the structure of PWA models is exploited to develop Rao-Blackwellized versions of the sampling algorithms. Only the marginal distribution of the parameters defining the regressor-space partition is approximated offline (resp. online) through MCMC simulation (resp. particle filters), while the conditional distribution of the other parameters given the regressor-space partition is computed (resp. updated) analytically.

The main strengths of the proposed Bayesian inference framework are: (*i*) one-shot learning of the regressor-space partition and of the local affine models; (*ii*) derivation of the posterior distribution of the model parameters and of the predicted output; (*iii*) the possibility of processing large datasets with a reasonable computational time that increases linearly with the length of the training data sequence; (*iv*) an incremental version of the learning algorithm for processing streaming data.

The approach can be generalized to: (*i*) estimate piecewise-nonlinear models by simply manipulating the regressors/inputs through nonlinear basis functions (*e.g.*, polynomials); (*ii*) handle output noises with full positive definite covariance matrix by considering a Gaussian–Wishart distribution as a prior on the parameters Θ and on inverse of the noise covariance matrix.

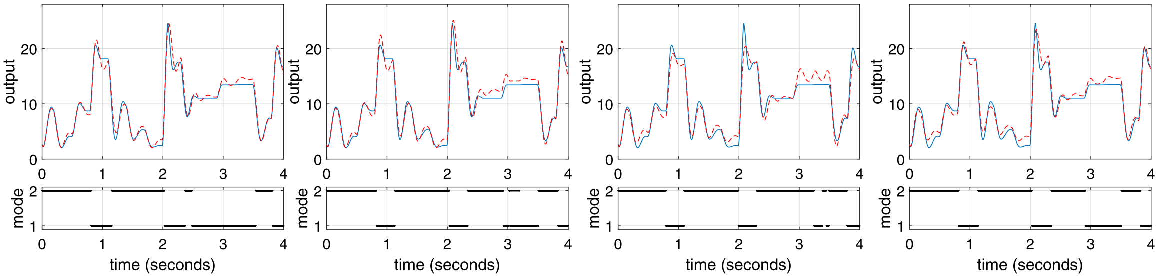
[5] The *Hybrid Identification Toolbox* (HIT) toolbox (Ferrari-Trecate, 2005) has been employed.



(a) Batch learning

(b) Recursive learning

Fig. 5. Pick and place machine: true output (blue line); estimated mean ± 3 standard deviation (grey region). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Batch learning

(b) Recursive learning

(c) Clustering approach [15]

(d) Opt.-based approach [4]

Fig. 6. Top: actual (blue line) and open-loop simulated output (red line); bottom: estimated mode sequence..

Appendix

A.1. Proof of Proposition 1

Before proving Proposition 1, a set of results useful for its detailed derivation is first provided.

Result 1. *Given three generic vectors $Y \in \mathbb{R}^N$, $\theta \in \mathbb{R}^{n_\theta}$ and $\bar{\theta} \in \mathbb{R}^{n_\theta}$, a matrix $X \in \mathbb{R}^{n_\theta, N}$, and two symmetric positive definite matrices $\Sigma_v \in \mathbb{R}^{N,N}$ and $\Sigma_\theta \in \mathbb{R}^{n_\theta, n_\theta}$, the following statements hold:*

1. *the expression*
$$e^{-\frac{1}{2}\left[(Y-X'\theta)'\Sigma_v^{-1}(Y-X'\theta)+(\theta-\bar{\theta})'\Sigma_\theta^{-1}(\theta-\bar{\theta})\right]} \tag{39}$$
is equal to
$$\begin{aligned}&(2\pi)^{\frac{n_\theta}{2}}|A|^{\frac{1}{2}}\times \\ &\times e^{-\frac{1}{2}\left(Y'\Sigma_v^{-1}Y+\bar{\theta}'\Sigma_\theta^{-1}\bar{\theta}-\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right)'\mu\right)}\mathcal{N}(\theta;\mu,A),\end{aligned} \tag{40}$$
with
$$A = (X\Sigma_v^{-1}X' + \Sigma_\theta^{-1})^{-1}, \;\; \mu = A\left(X\Sigma_v^{-1}Y + \Sigma_\theta^{-1}\bar{\theta}\right).$$
2. *the integral*
$$\int e^{-\frac{1}{2}\left[(Y-X'\theta)'\Sigma_v^{-1}(Y-X'\theta)+(\theta-\bar{\theta})'\Sigma_\theta^{-1}(\theta-\bar{\theta})\right]}d\theta$$
is equal to
$$(2\pi)^{\frac{n_\theta}{2}}|A|^{\frac{1}{2}}e^{-\frac{1}{2}\left(Y'\Sigma_v^{-1}Y+\bar{\theta}'\Sigma_\theta^{-1}\bar{\theta}-\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right)'\mu\right)}.$$

Proof. Let us consider the exponent in (39) up to the constant $-\frac{1}{2}$ and let us complete the square as follows

$$\begin{aligned}&\left(Y-X'\theta\right)'\Sigma_v^{-1}\left(Y-X'\theta\right)+(\theta-\bar{\theta})'\Sigma_\theta^{-1}(\theta-\bar{\theta}) \\ =&\theta'(X\Sigma_v^{-1}X'+\Sigma_\theta^{-1})\theta-2\theta'\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right) \\ &+Y'\Sigma_v^{-1}Y+\bar{\theta}'\Sigma_\theta^{-1}\bar{\theta} \\ =&\theta'A^{-1}\theta-2\theta'A^{-1}A\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right) \\ &+Y'\Sigma_v^{-1}Y+\bar{\theta}'\Sigma_\theta^{-1}\bar{\theta} \\ =&\left(\theta-A\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right)\right)'A^{-1}\left(\theta-A\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right)\right) \\ &-\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right)'A\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right) \\ &+Y'\Sigma_v^{-1}Y+\bar{\theta}'\Sigma_\theta^{-1}\bar{\theta}.\end{aligned}$$

Using the above equation, Eq. (39) can be rewritten as

$$\begin{aligned}&e^{-\frac{1}{2}\left(Y'\Sigma_v^{-1}Y+\bar{\theta}'\Sigma_\theta^{-1}\bar{\theta}-\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right)'\mu\right)}e^{-\frac{1}{2}(\theta-\mu)'A^{-1}(\theta-\mu)} \\ =&(2\pi)^{\frac{n_\theta}{2}}|A|^{\frac{1}{2}}e^{-\frac{1}{2}\left(Y'\Sigma_v^{-1}Y+\bar{\theta}'\Sigma_\theta^{-1}\bar{\theta}-\left(X\Sigma_v^{-1}Y+\Sigma_\theta^{-1}\bar{\theta}\right)'\mu\right)}\mathcal{N}(\theta;\mu,A).\end{aligned}$$

This proves the first statement. The second statement simply follows by integrating (40) w.r.t. θ. ■

Proposition 1 is now proved. Using Bayes' rule, let us rewrite the conditional distribution $p(\Theta, \sigma_v^{-2}|\Omega, X, Y)$ as

$$p(\Theta,\sigma_v^{-2}|\Omega,X,Y)=\frac{p(Y|\Theta,\sigma_v^{-2},\Omega,X)p(\Theta,\sigma_v^{-2}|\Omega)}{p(Y|\Omega,X)} \tag{41a}$$

$$=\frac{p(Y|\Theta,\sigma_v^{-2},\Omega,X)p(\Theta|\sigma_v^{-2})p(\sigma_v^{-2})}{\int p(Y|\Theta,\sigma_v^{-2},\Omega,X)p(\Theta|\sigma_v^{-2})p(\sigma_v^{-2})d\Theta d\sigma_v^{-2}} \tag{41b}$$

Based on the priors (4) and (5) on the parameters (Θ, σ_v^{-2}), and the likelihood in (8), the numerator in (41b) becomes

$$p(Y|\Theta,\Omega,\sigma_v^{-2},X)p(\Theta|\sigma_v^{-2})p(\sigma_v^{-2}) \tag{42a}$$

$$=\frac{(\sigma_v^{-2})^{\frac{n_y(T+n_\theta s)}{2}}}{(2\pi)^{\frac{n_y(T+n_\theta s)}{2}}}(\lambda^{-2})^{\frac{n_y n_\theta s}{2}}\Gamma\left(\sigma_v^{-2};\alpha_0,\beta_0\right) \tag{42b}$$

$$\times\prod_{i=1}^{s}\prod_{j=1}^{n_y}e^{-\frac{1}{2}\sigma_v^{-2}\left(\left(\mathbb{Y}_i^{(j)}-\mathbb{X}_i'\theta_i^{(j)}\right)'\left(\mathbb{Y}_i^{(j)}-\mathbb{X}_i'\theta_i^{(j)}\right)+\lambda^{-2}\theta_i^{(j)'}\theta_i^{(j)}\right)} \tag{42c}$$

Then, based on Result 1 [part 1], Eq. (42) is written as

$$
\begin{aligned}
&p(Y|\Theta,\Omega,\sigma_v^{-2},X)p(\Theta|\sigma_v^{-2})p(\sigma_v^{-2})\\
=&\frac{(\sigma_v^{-2})^{\frac{n_y(T+n_\theta s)}{2}}}{(2\pi)^{\frac{n_y(T+n_\theta s)}{2}}}(\lambda^{-2})^{\frac{n_y n_\theta s}{2}}\,\Gamma\left(\sigma_v^{-2};\alpha_0,\beta_0\right)\\
&\times\,(2\pi)^{\frac{n_y n_\theta s}{2}}\prod_{i=1}^{s}|\sigma_v^2 F_i|^{\frac{n_y}{2}}\\
&\times\prod_{i=1}^{s}\prod_{j=1}^{n_y}e^{-\frac{1}{2}\sigma_v^{-2}\left(\mathbb{Y}_i^{(j)'}\mathbb{Y}_i^{(j)}-\mathbb{Y}_i^{(j)'}\mathbb{X}_i'\mu_i^{(j)}\right)}\mathcal{N}(\theta_i^{(j)};\mu_i^{(j)},\sigma_v^2F_i),
\end{aligned}
$$

with F_i and μ_i defined in (12b) and (12c). Using the definition of *Gamma* distributions and simple algebraic manipulations, Eq. (44) is also equal to

$$
\begin{aligned}
&p(Y|\Theta,\Omega,\sigma_v^{-2},X)p(\Theta|\sigma_v^{-2})p(\sigma_v^{-2})\\
=&\frac{(\lambda^{-2})^{\frac{n_y n_\theta s}{2}}}{(2\pi)^{\frac{n_y T}{2}}}\prod_{i=1}^{s}|\mathbb{X}_i\mathbb{X}_i'+\lambda^{-2}I_{n_\theta}|^{-\frac{n_y}{2}}\prod_{i,j=1}^{s,n_y}\mathcal{N}(\theta_i^{(j)};\mu_i^{(j)},\sigma_v^2F_i)\\
&\times\frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)}\underbrace{(\sigma_v^{-2})^{\alpha_0+\frac{n_yT}{2}-1}e^{-\sigma_v^{-2}\left(\beta_0+\frac{1}{2}\sum_{i,j=1}^{s,n_y}\left(\mathbb{Y}_i^{(j)'}\mathbb{Y}_i^{(j)}-\mathbb{Y}_i^{(j)'}\mathbb{X}_i'\mu_i^{(j)}\right)\right)}}_{\frac{\Gamma(\alpha)}{\beta^\alpha}\Gamma(\sigma_v^{-2};\alpha,\beta)} \qquad (44)
\end{aligned}
$$

with α and β in (12d) and (12e), respectively.

The denominator in (41b) can be obtained by integrating (44) w.r.t. $\theta_i^{(j)}$ and σ_v^{-2}, thus obtaining

$$
\begin{aligned}
&p(Y|\Omega,X)=\int p(Y|\Theta,\sigma_v^{-2},\Omega,X)p(\Theta|\sigma_v^{-2})p(\sigma_v^{-2})d\Theta d\sigma_v^{-2}\\
=&\frac{(\lambda^{-2})^{\frac{n_y n_\theta s}{2}}}{(2\pi)^{\frac{n_y T}{2}}}\prod_{i=1}^{s}|\mathbb{X}_i\mathbb{X}_i'+\lambda^{-2}I_{n_\theta}|^{-\frac{n_y}{2}}\frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)}\frac{\Gamma(\alpha)}{\beta^\alpha}. \qquad (45)
\end{aligned}
$$

Taking the ratio of (44) to (45), Proposition 1 follows.

A.2. Proof of Proposition 2

Proposition 2 follows by rewriting $\dfrac{p(\Omega^*|X,Y)}{p(\Omega[k]|X,Y)}$ as

$$
\frac{p(\Omega^*|X,Y)}{p(\Omega[k]|X,Y)}=\frac{p(Y|\Omega^*,X)p(\Omega^*)}{p(Y|\Omega[k],X)p(\Omega[k])} \qquad (46)
$$

and by substituting the analytical expression of $p(Y|\Omega,X)$ provided in Eq. (45).

A.3. Proof of Proposition 3

Based on the modelling assumptions (1) and the conditional posterior distribution $p(\Theta|\sigma_v^{-2},\Omega,X,Y)$ in (12a), the distribution $p(y^\star|x^\star,\Omega[k],\sigma_v^{-2},X,Y)$ is given by

$$
\begin{aligned}
&p(y^\star|x^\star,\Omega[k],\sigma_v^{-2},X,Y)= \qquad (47a)\\
=&\mathcal{N}\left(y^\star;\mu_{i^\star}'[k]x^\star,\sigma_v^2(x^{\star'}(\mathbb{X}_{i^\star}[k]\mathbb{X}_{i^\star}'[k]+\lambda^{-2}I_{n_\theta})^{-1}x^\star+1)I_{n_y}\right). \qquad (47b)
\end{aligned}
$$

Furthermore, from Proposition 1, we have that

$$
p(\sigma_v^{-2}|\Omega[k],X,Y)=\Gamma(\sigma_v^{-2};\alpha,\beta[k]). \qquad (48)
$$

By combining Eq. (47) and (48), one obtains:

$$
\begin{aligned}
&p(y^\star|x^\star,\Omega[k],X,Y) \qquad (49a)\\
=&\int p(y^\star|x^\star,\Omega[k],\sigma_v^{-2},X,Y)p(\sigma_v^{-2}|\Omega[k],X,Y)d\sigma_v^{-2} \qquad (49b)\\
\propto&\int(\sigma_v^{-2})^{\frac{n_y}{2}+\alpha-1}e^{-\sigma_v^{-2}[\beta[k]+\frac{1}{2}(y^\star-\bar{y}[k])'H_{i^\star}^{-1}[k](y^\star-\bar{y}[k])]}d\sigma_v^{-2} \qquad (49c)
\end{aligned}
$$

with $H_{i^\star}[k]=(x^{\star'}(\mathbb{X}_{i^\star}[k]\mathbb{X}_{i^\star}'[k]+\lambda^{-2}I_{n_\theta})^{-1}x^\star+1)I_{n_y}$ and $\bar{y}[k]=\mu_{i^\star}'[k]x^\star$.

Note that term inside the integral in (49c) is a *Gamma* distribution of σ^{-2} up to the normalizing factor

$$
\frac{\Gamma(\frac{n_y}{2}+\alpha)}{\left(\beta[k]+\frac{1}{2}(y^\star-\mu_{i^\star}[k]'x^\star)'H_{i^\star}^{-1}[k](y^\star-\mu_{i^\star}'[k]x^\star)\right)^{\frac{n_y}{2}+\alpha}}
$$

The integral in (49c) can be easily computed and Eq. (49) becomes

$$
\begin{aligned}
&p(y^\star|x^\star,\Omega[k],X,Y)\\
\propto&\left(1+\frac{1}{2\beta[k]}(y^\star-\mu_{i^\star}'[k]x^\star)'H_{i^\star}^{-1}[k](y^\star-\mu_{i^\star}'[k]x^\star)\right)^{-\frac{n_y+2\alpha}{2}}\\
=&\left(1+\frac{1}{2\alpha}(y^\star-\mu_{i^\star}'[k]x^\star)'V_{i^\star}^{-1}[k](y^\star-\mu_{i^\star}'[k]x^\star)\right)^{-\frac{n_y+2\alpha}{2}}\\
=&St(y^\star;\mu_{i^\star}'[k]x^\star,V_{i^\star}[k],2\alpha),
\end{aligned}
$$

with $V_{i^\star}[k]=H_{i^\star}[k]\frac{\beta[k]}{\alpha}$ as in (16).

A.4. Proof of Proposition 4

For given $\Omega[k]$, the term $\max\limits_{\Theta,\sigma_v^{-2}}p(\Theta,\sigma_v^{-2}|\Omega[k],X,Y)$ can be computed analytically based on the expression of $p(\Theta,\sigma_v^{-2}|\Omega[k],X,Y)$ in Eq. (12a). Specifically, the maximum over Θ and σ_v^{-2} of $p(\Theta,\sigma_v^{-2}|\Omega[k],X,Y)$ is achieved for

$$
\theta_i=\mu_i[k],\quad \sigma_v^{-2}=\frac{\alpha+\frac{sn_yn_\theta}{2}-1}{\beta[k]}. \qquad (51)
$$

Substitution of (51) into the definition of $p(\Theta,\sigma_v^{-2}|\Omega[k],X,Y)$ (Eq. (12a)) leads to

$$
\begin{aligned}
&\max_{\Theta,\sigma_v^{-2}}p(\Theta,\sigma_v^{-2}|\Omega[k],X,Y)\\
=&\frac{\beta[k]}{\Gamma(\alpha)}(\alpha+\frac{sn_yn_\theta}{2}-1)^{\alpha-1}\frac{\left(\frac{\alpha+\frac{sn_yn_\theta}{2}-1}{\beta[k]}\right)^{\frac{sn_yn_\theta}{2}}}{(2\pi)^{\frac{sn_yn_\theta}{2}}}e^{1-\alpha-\frac{sn_yn_\theta}{2}}\\
&\times\prod_{i=1}^{s}|\mathbb{X}_i[k]\mathbb{X}_i'[k]+\lambda^{-2}I_{n_\theta}|^{\frac{n_y}{2}}. \qquad (52)
\end{aligned}
$$

Substituting (52) into Eq. (18) and ignoring the terms which do not depend on the partition $\mathcal{X}[\Omega[k]]$, we obtain

$$
\begin{aligned}
\hat{\Omega}=&\operatorname*{argmax}_{\{\Omega[k]\}_{k=1}^M}p(\Omega[k]|X,Y)(\beta[k])^{1-\frac{sn_yn_\theta}{2}}\\
&\qquad\times\prod_{i=1}^{s}|\mathbb{X}_i[k]\mathbb{X}_i'[k]+\lambda^{-2}I_{n_\theta}|^{\frac{n_y}{2}}\\
=&\operatorname*{argmax}_{\{\Omega[k]\}_{k=1}^M}p(Y|\Omega[k],X)p(\Omega[k])(\beta[k])^{1-\frac{sn_yn_\theta}{2}}\\
&\qquad\times\prod_{i=1}^{s}|\mathbb{X}_i[k]\mathbb{X}_i'[k]+\lambda^{-2}I_{n_\theta}|^{\frac{n_y}{2}}
\end{aligned}
$$

By substituting the expression of the marginal likelihood $p(Y|\Omega[k],X)$ (Eq. (45)) into the above equation, Eq. (19) follows. Eq. (20) follows from (51) and by constructing the parameters μ_i and β based on the partition $\mathcal{X}[\hat{\Omega}]$.

A.5. Proof of Proposition 5

Eq. (29a) follows from the definition of $F_{i,t}$ in (28a) and the construction of the regressor matrix $\mathbb{X}_{i,t}$ in (26b). Eq. (29b) follows by applying the *Matrix Inversion Lemma* to Eq. (29a).

Eq. (29c) can be derived from the following algebraic manipulations

$$\begin{aligned}\mu_{i,t} =& F_{i,t}\mathbb{X}_{i,t}\mathbb{Y}_{i,t} = F_{i,t}\left(\mathbb{X}_{i,t-1}\mathbb{Y}_{i,t-1} + \left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right] y_t'\mathbb{I}(s_{t|t}=i)\right)\\ =& F_{i,t}\left(F_{i,t-1}^{-1}\mu_{i,t-1} + \left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right] y_t'\mathbb{I}(s_{t|t}=i)\right)\\ =& \mu_{i,t-1} + F_{i,t}\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right](y_t' - \left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{i,t-1})\mathbb{I}(s_{t|t}=i).\end{aligned}$$

As for Eq. (29d), we have

$$\begin{aligned}\beta_t =& \beta_0 + \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}\left(\mathbb{Y}_{i,t}^{(j)'}\mathbb{Y}_{i,t}^{(j)} - \mathbb{Y}_{i,t}^{(j)'}\mathbb{X}_{i,t}'\mu_{i,t}^{(j)}\right)\\ =& \beta_0 + \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}\mathbb{Y}_{i,t-1}^{(j)'}\mathbb{Y}_{i,t-1}^{(j)} + y_t^{(j)}y_t^{(j)}\mathbb{I}(s_{t|t}=i)\\ &- \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}\mathbb{Y}_{i,t-1}^{(j)'}\mathbb{X}_{i,t-1}'\mu_{i,t}^{(j)} + y_t^{(j)}\left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{i,t}^{(j)}\mathbb{I}(s_{t|t}=i)\\ =& \beta_{t-1} + \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}(y_t^{(j)}y_t^{(j)} - y_t^{(j)}\left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{i,t}^{(j)})\mathbb{I}(s_{t|t}=i)\\ &- \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{n_y}\mu_{i,t-1}^{(j)'}F_{i,t-1}^{-1}F_{i,t}\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]\\ &\times (y_t^{(j)} - \left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{i,t-1}^{(j)})\mathbb{I}(s_{t|t}=i) \qquad (54)\end{aligned}$$

By substituting $F_{i,t-1}^{-1} = F_{i,t}^{-1} - \left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & x_t\end{smallmatrix}\right]\mathbb{I}(s_{t|t} = i)$ into (54), Eq. (29d) follows.

A.6. Proof of Proposition 6

To simplify the notation, the index $[k]$ is dropped from: the particle's trajectory $\Omega_{1:t}[k]$; the active state $s_{t|t}[k]$; and the parameters $F_{i,t}[k]$ and $\beta_t[k]$. Furthermore, the conditional dependence of $p(y_t|\Omega_{1:t}[k], X_{1:t-1}, x_t, Y_{1:t-1})$ on the past input sequence $X_{1:t-1}$ is omitted. Let us rewrite the likelihood $p(y_t|\Omega_{1:t}, x_t, Y_{1:t-1})$ as

$$\int p(y_t|\Theta, \sigma_v^{-2}, \Omega_{1:t}, x_t, Y_{1:t-1})p(\Theta, \sigma_v^{-2}|\Omega_{1:t-1}, Y_{1:t-1})d\Theta d\sigma_v^{-2}. \qquad (55)$$

Note that the conditional likelihood $p(y_t|\Theta, \sigma_v^{-2}, \Omega_{1:t}, x_t, Y_{1:t-1})$ is equal to

$$p(y_t|\Theta, \sigma_v^{-2}, \Omega_t, x_t) = \mathcal{N}\left(y_t; \theta_{s_{t|t}}'\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right], \sigma_v^2 I_{n_y}\right). \qquad (56)$$

By substituting (27) and (56) into (55), we obtain

$$\begin{aligned}&\int p(y_t|\Theta, \sigma_v^{-2}, \Omega_{1:t}, x_t, Y_{1:t-1})p(\Theta, \sigma_v^{-2}|\Omega_{1:t-1}, Y_{1:t-1})d\Theta d\sigma_v^{-2}\\ =& \int \mathcal{N}\left(y_t; \theta_{s_{t|t}}'\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right], \sigma_v^2 I_{n_y}\right)\Gamma(\sigma_v^{-2}; \alpha_{t-1}, \beta_{t-1})\\ &\times \prod_{j=1}^{n_y}\mathcal{N}(\theta_{s_{t|t}}^{(j)}; \mu_{s_{t|t},t-1}^{(j)}, \sigma_v^2 F_{s_{t|t},t-1})d\theta_{s_{t|t}}^{(j)}d\sigma_v^{-2}.\end{aligned}$$

From definitions of *Normal* and *Gamma* distributions, the integral above reads

$$\begin{aligned}&\frac{1}{(2\pi)^{\frac{n_y+n_\theta n_y}{2}}}\frac{(\beta_{t-1})^{\alpha_{t-1}}}{\Gamma(\alpha_{t-1})}\left|F_{s_{t|t},t-1}\right|^{-\frac{n_y}{2}}\\ &\times\int(\sigma_v^{-2})^{\alpha_{t-1}+\frac{n_y+n_\theta n_y}{2}-1}e^{-\beta_{t-1}\sigma_v^{-2}}\\ &\times\prod_{j=1}^{n_y}e^{-\frac{1}{2}\sigma_v^{-2}(y_t^{(j)}-\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]'\theta_{s_{t|t}}^{(j)})'(y_t^{(j)}-\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]'\theta_{s_{t|t}}^{(j)})}\\ &\times\prod_{j=1}^{n_y}e^{-\frac{1}{2}\sigma_v^{-2}(\theta_{s_{t|t}}^{(j)}-\mu_{s_{t|t},t-1}^{(j)})'F_{s_{t|t},t-1}^{-1}(\theta_{s_{t|t}}^{(j)}-\mu_{s_{t|t},t-1}^{(j)})}d\theta_{s_{t|t}}^{(j)}d\sigma_v^{-2}. \qquad (57)\end{aligned}$$

Using Result 1 [part 2] to compute the above integral w.r.t. $\theta_{s_{t|t}}^{(j)}$ and noticing (from (29a) and (28b)) that

$$\underbrace{\left(\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right] + F_{s_{t|t},t-1}^{-1}\right)^{-1}}_{F_{s_{t|t},t}}\underbrace{\left(\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]y_t^{(j)} + F_{s_{t|t},t-1}^{-1}\mu_{s_{t|t},t-1}^{(j)}\right)}_{\mathbb{X}_{s_{t|t},t}\mathbb{Y}_{s_{t|t},t}} = \mu_{s_{t|t},t}^{(j)},$$

Eq. (57) then reads as follows

$$\begin{aligned}&\frac{1}{(2\pi)^{\frac{n_y}{2}}}\frac{(\beta_{t-1})^{\alpha_{t-1}}}{\Gamma(\alpha_{t-1})}\left|F_{s_{t|t},t-1}\right|^{-\frac{n_y}{2}}\left|\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right] + F_{s_{t|t},t-1}^{-1}\right|^{-\frac{n_y}{2}}\\ &\times\int(\sigma_v^{-2})^{\alpha_{t-1}+\frac{n_y}{2}-1}e^{-\beta_{t-1}\sigma_v^{-2}}\\ &\quad\times\prod_{j=1}^{n_y}e^{-\frac{1}{2}\sigma_v^{-2}\left(y_t^{(j)}y_t^{(j)}+\mu_{s_{t|t},t-1}^{(j)'}F_{s_{t|t},t-1}^{-1}\mu_{s_{t|t},t-1}^{(j)}\right)}\\ &\qquad\times e^{\frac{1}{2}\sigma_v^{-2}\left(\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]y_t^{(j)}+F_{s_{t|t},t-1}^{-1}\mu_{s_{t|t},t-1}^{(j)}\right)'\mu_{s_{t|t},t}^{(j)}}d\sigma_v^{-2} \qquad (58a)\\ =&\frac{1}{(2\pi)^{\frac{n_y}{2}}}\frac{(\beta_{t-1})^{\alpha_{t-1}}}{\Gamma(\alpha_{t-1})}\left|F_{s_{t|t},t-1}\right|^{-\frac{n_y}{2}}|\underbrace{\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right] + F_{s_{t|t},t-1}^{-1}}_{F_{s_{t|t},t}^{-1}}|^{-\frac{n_y}{2}}\\ &\quad\times\int(\sigma_v^{-2})^{\alpha_{t-1}+\frac{n_y}{2}-1}e^{-\beta_t\sigma_v^{-2}}d\sigma_v^{-2} \qquad (58b)\end{aligned}$$

where the last equation comes from the following algebraic manipulations of the exponents in (58a):

$$\begin{aligned}&\beta_{t-1} + \frac{1}{2}\sum_{j=1}^{n_y}\left(y_t^{(j)}y_t^{(j)} + \mu_{s_{t|t},t-1}^{(j)'}F_{s_{t|t},t-1}^{-1}\mu_{s_{t|t},t-1}^{(j)}\right)\\ &- \frac{1}{2}\sum_{j=1}^{n_y}\left(\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right]y_t^{(j)} + F_{s_{t|t},t-1}^{-1}\mu_{s_{t|t},t-1}^{(j)}\right)'\mu_{s_{t|t},t}^{(j)}\\ =&\beta_{t-1} + \frac{1}{2}\sum_{j=1}^{n_y}y_t^{(j)}y_t^{(j)} - y_t^{(j)}\left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{s_{t|t},t}^{(j)}\\ &- \frac{1}{2}\sum_{j=1}^{n_y}\left(\mu_{s_{t|t},t-1}^{(j)'}F_{s_{t|t},t-1}^{-1}F_{s_{t|t},t}\left[\begin{smallmatrix}1\\x_t\end{smallmatrix}\right](y_t^{(j)} - \left[\begin{smallmatrix}1 & x_t'\end{smallmatrix}\right]\mu_{s_{t|t},t-1}^{(j)})\right)\\ =&\beta_t.\end{aligned}$$

The last equation comes from (54). Going back to (58b) and noticing that the term in the integral is a *Gamma* distribution $\Gamma(\sigma_v^{-2}; \alpha_t, \beta_t)$ up to the scaling constant $\frac{(\beta_t)^{\alpha_t}}{\Gamma(\alpha_t)}$, Eq. (58b) can be easily solved and we finally obtain that $p(y_t|\Omega_{1:t}, x_t, Y_{1:t-1})$ is equal to

$$\begin{aligned}&\frac{1}{(2\pi)^{\frac{n_y}{2}}}\frac{(\beta_{t-1})^{\alpha_{t-1}}}{\Gamma(\alpha_{t-1})}\frac{\Gamma(\alpha_t)}{(\beta_t)^{\alpha_t}}\left|F_{s_{t|t},t-1}\right|^{-\frac{n_y}{2}}|F_{s_{t|t},t}^{-1}|^{-\frac{n_y}{2}}\\ \propto&\frac{(\beta_{t-1})^{\alpha_{t-1}}}{(\beta_t)^{\alpha_{t-1}}}\left|F_{s_{t|t},t-1}\right|^{-\frac{n_y}{2}}|F_{s_{t|t},t}^{-1}|^{-\frac{n_y}{2}}.\end{aligned}$$

A.7. Proof of Proposition 7

First, the maximization problem in (34) is factorized as

$$\begin{aligned}&\hat{\Theta}_t, \hat{\sigma}_{v,t}^{-2}, \hat{\Omega}_{1:t} =\\ =&\ \underset{\Theta,\sigma_v^{-2},\Omega_{1:t}}{\text{argmax}}\ p(\Omega_{1:t}|X_{1:t}, Y_{1:t})p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t})\\ =&\arg\max_{\Omega_{1:t}} p(\Omega_{1:t}|X_{1:t}, Y_{1:t})\max_{\Theta,\sigma_v^{-2}} p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t}). \qquad (59)\end{aligned}$$

For a given trajectory $\Omega_{1:t}$, the maximum of the conditional posterior $p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t})$ can be computed analytically from (27). Specifically, its maximum is achieved for

$$\theta_{i,t} = \mu_{i,t}[\Omega_{1:t}], \quad \sigma_{v,t}^{-2} = \frac{\alpha_t + \frac{sn_yn_\theta}{2} - 1}{\beta_t[\Omega_{1:t}]}. \qquad (60)$$

Substitution of the maximizers (60) into (27) leads to

$$\begin{aligned}
&\max_{\Theta,\sigma_v^{-2}} p(\Theta, \sigma_v^{-2}|\Omega_{1:t}, X_{1:t}, Y_{1:t}) \\
=&\frac{\beta_t[\Omega_{1:t}]}{\Gamma(\alpha_t)}(\alpha_t + \frac{sn_y n_\theta}{2} - 1)^{\alpha_t - 1}\frac{(\frac{\alpha_t+\frac{sn_yn_\theta}{2}-1}{\beta_t[\Omega_{1:t}]})^{\frac{sn_yn_\theta}{2}}}{(2\pi)^{\frac{sn_yn_\theta}{2}}} \\
&\times\ e^{1-\alpha_t-\frac{sn_yn_\theta}{2}}\prod_{i=1}^{s}|F_{i,t}^{-1}[\Omega_{1:t}]|^{\frac{ny}{2}}. \qquad (61)
\end{aligned}$$

By substituting (61) into (59) and by removing the terms which do not depend on $\Omega_{1:t}$, Eq. (35a) follows.

Eq. (35b) follows by substituting the MAP estimate $\hat{\Omega}_{1:t}$ into (60).

A.8. Proof of Proposition 8

Let us rewrite $\tilde{p}(\Omega_{1:t}|X_{1:t}, Y_{1:t})$ as

$$\begin{aligned}
&\tilde{p}(\Omega_{1:t}|X_{1:t}, Y_{1:t}) = p(\Omega_{1:t}|X_{1:t}, Y_{1:t})\beta_t^{1-\frac{sn_yn_\theta}{2}}\prod_{i=1}^{s}|F_{i,t}^{-1}|^{\frac{ny}{2}} \\
&= p(y_t|\Omega_{1:t}, X_{1:t}, Y_{1:t-1})\beta_t^{1-\frac{sn_yn_\theta}{2}}\prod_{i=1}^{s}|F_{i,t}^{-1}|^{\frac{ny}{2}} \\
&\times\ p(\Omega_t|\Omega_{t-1})p(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1}). \\
&= p(y_t|\Omega_{1:t}, X_{1:t}, Y_{1:t-1})\frac{\beta_t^{1-\frac{sn_yn_\theta}{2}}}{\beta_{t-1}^{1-\frac{sn_yn_\theta}{2}}}\frac{\prod_{i=1}^{s}|F_{i,t}^{-1}|^{\frac{ny}{2}}}{\prod_{i=1}^{s}|F_{i,t-1}^{-1}|^{\frac{ny}{2}}} \\
&\times\ p(\Omega_t|\Omega_{t-1})\tilde{p}(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1}). \qquad (62)
\end{aligned}$$

Then, from Proposition 6 and Eq. (62), we obtain

$$\begin{aligned}
&\tilde{p}(\Omega_{1:t}|X_{1:t}, Y_{1:t}) \\
\propto&\frac{(\beta_{t-1})^{\alpha_{t-1}}}{(\beta_t)^{\alpha_t}}\left|F_{s_{t|t},t-1}\right|^{-\frac{ny}{2}}\left|F_{s_{t|t},t}^{-1}\right|^{-\frac{ny}{2}} \\
&\times\ \frac{(\beta_t)^{1-\frac{sn_yn_\theta}{2}}}{(\beta_{t-1})^{1-\frac{sn_yn_\theta}{2}}}\frac{\prod_{i=1}^{s}|F_{i,t}^{-1}|^{\frac{ny}{2}}}{\prod_{i=1}^{s}|F_{i,t-1}^{-1}|^{\frac{ny}{2}}} \\
&\times\ p(\Omega_t|\Omega_{t-1})\tilde{p}(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1}) \\
=&\frac{(\beta_t)^{1-\frac{sn_yn_\theta}{2}-\alpha_t}}{(\beta_{t-1})^{1-\frac{sn_yn_\theta}{2}-\alpha_{t-1}}}p(\Omega_t|\Omega_{t-1})\tilde{p}(\Omega_{1:t-1}|X_{1:t-1}, Y_{1:t-1}).
\end{aligned}$$

This completes the proof.

References

Andrieu, C., de Freitas, N., Doucet, A., & Jordan, M. I. (2003). An introduction to MCMC for machine learning. *Machine Learning*, *50*(1–2), 5–43.

Bako, L. (2011). Identification of switched linear systems via sparse optimization. *Automatica*, *47*(4), 668–677.

Bako, L., Boukharouba, K., Duviella, E., & Lecoeuche, S. (2011). A recursive identification algorithm for switched linear/affine models. *Nonlinear Analysis. Hybrid Systems*, *5*(2), 242–253.

Bemporad, A., Breschi, V., Piga, D., & Boyd, S. (2018). Fitting jump models. *Automatica*, *96*, 11–21.

Bemporad, A., Ferrari-Trecate, G., & Morari, M. (2000). Observability and controllability of piecewise affine and hybrid systems. *IEEE Transactions on Automatic Control*, *45*(10), 1864–1876.

Bemporad, A., Garulli, A., Paoletti, S., & Vicino, A. (2005). A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control*, *50*(10), 1567–1580.

Bemporad, A., & Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, *35*(3), 407–427.

Bennett, K. P., & Mangasarian, O. L. (1994). Multicategory discrimination via linear programming. *Optimization Methods & Software*, *3*(1–3), 27–39.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Breiman, L. (1993). Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory*, *39*(3), 999–1013.

Breschi, V., Piga, D., & Bemporad, A. (2016). Piecewise affine regression via recursive multiple least squares and multicategory discrimination. *Automatica*, *73*, 155–162.

Casella, G., & Robert, C. P. (1996). Rao-blackwellisation of sampling schemes. *Biometrika*, *83*(1), 81–94.

Chib, S., & Greenberg, E. (1995). Understanding the metropolis-hastings algorithm. *The American Statistician*, *49*(4), 327–335.

Ferrari-Trecate, G. (2005). *Hybrid identification toolbox*.

Ferrari-Trecate, G., Muselli, M., Liberati, D., & Morari, M. (2003). A clustering technique for the identification of piecewise affine systems. *Automatica*, *39*(2), 205–217.

Heemels, W., De Schutter, B., & Bemporad, A. (2001). Equivalence of hybrid dynamical models. *Automatica*, *37*(7), 1085–1091.

Juloski, A., Heemels, W., & Ferrari-Trecate, G. (2004). Data-based hybrid modelling of the component placement process in pick-and-place machines. *Control Engineering Practice*, *12*(10), 1241–1252.

Juloski, A. L., Weiland, S., & Heemels, W. (2005). A Bayesian approach to identification of hybrid systems. *IEEE Transactions on Automatic Control*, *50*(10), 1520–1533.

Lauer, F. (2015). On the complexity of piecewise affine system identification. *Automatica*, *62*, 148–153.

Naik, V. V., Mejari, M., Piga, D., & Bemporad, A. (2017). Regularized moving-horizon piecewise affine regression using mixed-integer quadratic programming. In *25th mediterranean conference on control and automation* (pp. 1349–1354). Valletta, Malta.

Ohlsson, H., & Ljung, L. (2013). Identification of switched linear regression models using sum-of-norms regularization. *Automatica*, *49*(4), 1045–1050.

Piga, D., & Tóth, R. (2013). An SDP approach for ℓ_0-minimization: Application to ARX model segmentation. *Automatica*, *49*(12), 3646–3653.

Pillonetto, G. (2016). A new kernel-based approach to hybrid system identification. *Automatica*, *70*, 21–31.

Roll, J., Bemporad, A., & Ljung, L. (2004). Identification of piecewise affine systems via mixed-integer programming. *Automatica*, *40*(1), 37–50.

Wågberg, J., Lindsten, F., & Schön, T. B. (2015). Bayesian nonparametric identification of piecewise affine ARX systems. *IFAC-PapersOnLine*, *48*(28), 709–714.

Dario Piga received his Ph.D. in Systems Engineering from the Politecnico di Torino (Italy) in 2012. He was a Postdoctoral Researcher at the Delft University of Technology (The Netherlands) in 2012 and at the Eindhoven University of Technology (The Netherlands) in 2013. From 2014 to early 2017 he was Assistant Professor at the IMT School for Advanced Studies Lucca (Italy) and since March 2017 he has been Senior Researcher at the IDSIA Dalle Molle Institute for Artificial Intelligence in Lugano (Switzerland) and Lecturer at the SUPSI University of Applied Sciences and Arts of Southern Switzerland. His main research interests include system identification, robust control, Bayesian filtering and non-convex optimization, with applications to process control and smart manufacturing.

Alberto Bemporad received his Master's degree in Electrical Engineering in 1993 and his Ph.D. in Control Engineering in 1997 from the University of Florence, Italy. In 1996/97 he was with the Center for Robotics and Automation, Department of Systems Science & Mathematics, Washington University, St. Louis. In 1997–1999 he held a postdoctoral position at the Automatic Control Laboratory, ETH Zurich, Switzerland, where he collaborated as a senior researcher until 2002. In 1999–2009 he was with the Department of Information Engineering of the University of Siena, Italy, becoming an Associate Professor in 2005. In 2010–2011 he was with the Department of Mechanical and Structural Engineering of the University of Trento, Italy. Since 2011 he is Full Professor at the IMT School for Advanced Studies Lucca, Italy, where he served as the Director of the institute in 2012–2015.

He spent visiting periods at Stanford University, University of Michigan, and Zhejiang University. In 2011 he cofounded ODYS S.r.l., a company specialized in developing model predictive control systems for industrial production. He has published more than 350 papers in the areas of model predictive control, hybrid systems, optimization, automotive control, and is the co-inventor of 16 patents. He is author or coauthor of various MATLAB toolboxes for model predictive control design, including the Model Predictive Control Toolbox (The Mathworks, Inc.) and the Hybrid Toolbox. He was an Associate Editor of the IEEE Transactions on Automatic Control during 2001–2004 and Chair of the Technical Committee on Hybrid Systems of the IEEE Control Systems Society in 2002–2010. He received the IFAC High-Impact Paper Award for the 2011–2014 triennial and the IEEE CSS Transition to Practice Award in 2019. He is an IEEE Fellow since 2010.

Alessio Benavoli received his Master's degree (2004) and his Ph.D. (2008) in Computer and Control Engineering from the University of Florence, Italy. From 2007 to 2008, he worked for the international company SELEX-Sistemi Integrati as system analyst. From 2008 to 2019, he was at the Dalle Molle Institute for Artificial Intelligence (IDSIA) in Lugano, Switzerland, becoming professor in 2018. He is currently Senior Lecturer at the Department of Computer Science and Information Systems (CSIS), University of Limerick, Ireland. His research interests are in the areas of probabilistic AI, Bayesian nonparametrics, and state estimation for dynamical systems. He has co-authored about 90 peer reviewed publications.