A Gaussian Process Model for Data Association and a Semi-Definite Programming Solution

Miguel Lázaro-Gredilla, *Member, IEEE,* and Steven Van Vaerenbergh, *Member, IEEE*

***Abstract*—In this paper we propose a Bayesian model for the data association problem, in which trajectory smoothness is enforced through the use of Gaussian process priors. This model allows to score candidate associations by using the evidence framework, thus casting the data association problem into an optimization problem. Under some additional mild assumptions, this optimization problem is shown to be equivalent to a constrained Max K-Section problem. Furthermore, for $K = 2$, a MaxCut formulation is obtained, to which an approximate solution can be efficiently found using an SDP relaxation. Solving this MaxCut problem is equivalent to finding the optimal association out of the combinatorially many possibilities. The obtained clustering depends only on two hyperparameters, which can also be selected by maximum evidence.**

***Index Terms*—clustering, data association, Gaussian processes, multi-target tracking, semi-definite programming**

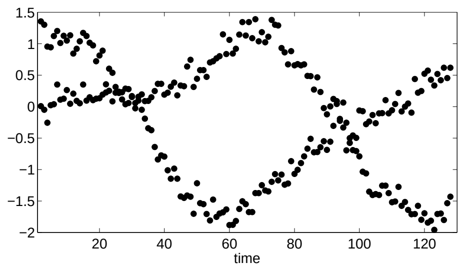


Fig. 1. A data association problem with two moving sources, in which one observation (black dots) from each source is measured at each time instant. Data association aims to determine which observations, measured at different time instants, originate from each source.

I. Introduction

Data association is a fundamental problem in multi-target tracking and computer vision. Given a set of observations that represent the positions of a number of sources, typically in motion, such as persons, vehicles or particles, data association consists in inferring which observations originate from each source[1] [1], [2]. For instance, given a set of unlabeled measurements of the positions of two persons in a room at several time instants, we would like to group the measurements that correspond to each person. Fig. 1 illustrates this concept with a typical example including two moving particles in a one-dimensional setting. Data association is encountered in many target tracking applications, such as sensor networks [3], radar tracking [4], and computer vision [5], and it is sometimes referred to as motion correspondence [2], or multiple model regression (MME) [6] in case motion is not explicitly considered.

Traditional multi-target tracking algorithms operate online. To predict the target's trajectory they use tracking techniques that incorporate the knowledge of the motion model, such as joint Kalman filters [7] or joint particle filters [8]. Given the predicted positions of the targets and a number of candidate observed positions, they commonly make instant data association decisions based on statistical approaches such as the Joint Probabilistic Data-Association Filter (JPDAF) [4], [8] and the Multiple Hypothesis Tracker (MHT) [7]. An important disadvantage of these techniques is that they usually require a large number of parameters, which motivated the development of several conceptually simpler approaches based on motion geometry heuristics [2], [9], [10]. These approaches are usually limited to specific scenarios, and they show difficulties in the presence of a large amount of noise or when several trajectories cross. Another strategy to improve performance relies on postponing the decisions until enough information is available to exclude ambiguities [2], although this causes the number of possible trajectories to grow exponentially. Several attempts have been made to restrain this combinatorial explosion, including [11], [12], [13].

We present an approach based on Gaussian processes (GPs) [14] that is capable of tackling these issues. Specifically, it models the target trajectories as GPs and it only requires to determine two hyperparameters, i.e. an approximate estimate of the Signal-to-Noise ratio (SNR) and a timescale, which relates to the motion smoothness. Furthermore, it is able to consider all available data points in batch form whilst avoiding the exponential growth in potential tracks. Additionally, it does not require the time instants to be uniformly spaced, and it can even be applied to problems where the input space has multiple dimensions. On the other hand, the presented algorithm requires that at each time instant all K sources are observed, resulting in K measurements per time instant. In other words, in this work we do not explicitly deal with the problem of missing data. We provide an efficient implementation for the case of $K = 2$ sources (such as the case of Fig. 1).

Gaussian processes are a powerful tool for Bayesian nonlinear regression due to the way in which they naturally incorporate smoothness [15]. In this case we will take advantage of this smoothness to model trajectories. Since there will be

M. Lázaro-Gredilla is with the Dept. of Signal Theory and Communications, University Carlos III in Madrid, Spain, Email: miguel@tsc.uc3m.es.

S. Van Vaerenbergh is with the Dept. of Communications Engineering, University of Cantabria, Spain, Email: steven@gtas.dicom.unican.es.

This work was supported by MICINN (Spanish Ministry for Science and Innovation) under grants TEC2010-19545-C04-03 (COSIMA) and CONSOLIDER-INGENIO 2010 CSD 2008-00010 (COMONSENS).

[1]We will use the terms source and target interchangeably.

multiple targets for any given location in the input space, typically corresponding to multiple objects at the same time instant in a tracking system, we will use a GP *mixture* in which each component is used to model one trajectory. Standard GP mixtures are used to model data with local nonstationarities or discontinuities [16], [17], [18], [19], but they are not designed to deal with multiple trajectories. This issue was recently tackled in [13], which introduced an *overlapping* mixture of Gaussian processes (OMGP) and proposed an approximate variational Bayesian solution involving a nonlinear optimization over a large number of free parameters.

The present work addresses the data association problem by modeling trajectories as independent GPs, just as [13]. The novelties of this paper with respect to [13] are:

- The non-linear optimization problem posed by the GP model is addressed in [13] by using variational inference to obtain an approximate posterior. This requires non-linear minimization of a non-convex objective. Thus, different initializations of the algorithm, as well as different non-linear optimization techniques, may result in different solutions. I.e., the algorithm is bound to get trapped in local minima (which may be very poor for difficult problems).
  In contrast, the present approach relaxes the original minimization objective to a convex problem that can be optimized very efficiently using semidefinite programming. This optimization process is guaranteed to always converge to the unique, global optimum. Then, using the random hyperplane rounding technique (see Section V), we retrieve a solution for the original problem that is guaranteed to be very close to the global optimum. Such guarantee was not available in [13], which could possibly get stuck at very bad local minima.
- Because we are minimizing a convex functional, the minimization process is much faster than in [13]. In Fig. 4 we include a comparison of the running times for GP-BTT and OMGP, for different numbers of observations. GP-BTT is remarkably faster than OMGP, typically a factor of 10 or greater.
- The variational algorithm provided by [13] works for any number K of trajectories. In contrast, the algorithm provided in this work is only valid for $K = 2$ trajectories and is casted as a MaxCut optimization problem. Despite this limitation, in Section V-A we show that for a general number of trajectories K, it is possible to re-cast the problem as a standard Max K-Section problem plus an additional linear constraint. We conjecture that it is also possible to solve this problem using semidefinite programming, thus paving the way for the development of a more general technique for the cases in which $K > 2$.

The remainder of this paper is organized as follows. The data association problem is shortly described in Section II. In Section III, an introduction to Gaussian process regression is given, after which a GP-based model for data association is proposed in Section IV. Section V details how the model is reduced to a Max K-Section problem by considering only one observation per source per time instant, and in Section VI an efficient algorithm for the case of $K = 2$ is proposed. Finally, a number of experiments are conducted in Section VII, and Section VIII summarizes the main conclusions of this work.

II. Problem setting

We start by formally describing the data association problem. Consider K independent data sources that generate a total of n observations. Each observation is generated by a source $s_i \in \{1, \dots, K\}$, with $i = 1, \dots, n$, and corresponds to a sampling location in the input space. Throughout the discussion we will use a one-dimensional input space that represents time, in order to focus on the target-tracking application, although the proposed method is not limited to one-dimensional input spaces. The i-th observation is thus taken at a time instant $t_i \in \mathbb{R}$, and it consists of D components, collected in a vector observation $[y_1(t_i), \dots, y_D(t_i)]^\top \in \mathbb{R}^D$. In the initial problem description we do not assume any additional restrictions. In particular, it is not required to receive an observation from each source at each time instant, and several observations, either from the same or different sources, can occur at the same instant (i.e., for $i \neq j$ it is possible to have $t_i = t_j$). Given all observations, the data association problem consists in finding which source corresponds to each observation.

In this work we assume that each source $k \in \{1, \dots, K\}$ is fully described by a set of (unobservable) latent functions $\{f_d^k(\cdot)\}_{d=1}^D$, which can be interpreted as trajectories in a D-dimensional space. Furthermore, each observation $y_d(t_i)$ is regarded as a noisy version of the corresponding latent function $f_d^k(t_i)$. Note how the unobservable latent functions carry a super-index k to indicate which source it corresponds to, whereas observations do not carry such an index, since this labeling is the very purpose of the data association problem.

Our objective is two-fold: We want to a) approximately recover the set of labels $\mathbf{s} = [s_1, \dots, s_n]^\top$ that identify the source of each data point (i.e., solve the data association problem) from observable data $\mathcal{D} \equiv \{t_i, y_1(t_i), \dots, y_D(t_i)\}_{i=1}^n$, and b) infer the latent function describing each source $\{f_d^k(\cdot)\}_{d=1}^D$, so that we can predict the outputs of the sources at new time instants.

III. Background on Gaussian processes for regression

Within the Bayesian framework, Gaussian processes (GPs) are a type of stochastic process commonly used as a prior for functions. The defining property of GPs is that any finite collection of its samples forms a multivariate Gaussian random variable. They have recently attracted a lot of attention due to their nice analytical properties and their state-of-the-art performance in regression tasks (see [15]). In this work, we will use independent GP priors on the functions that describe each trajectory over each dimension. Some background on GPs and its use for regression is provided here.

Assuming that a set of time-observation pairs $\mathcal{D} \equiv \{t_i, y(t_i)\}_{i=1}^n$ corresponding to a single trajectory are available, the regression task goal is, given a new input t_*, obtain a predictive distribution for the corresponding observation $y(t_*)$ based on $\mathcal{D}$.

The GP regression model assumes that the observations can be modeled as some noiseless latent function of time plus independent noise

$$y = f(t) + \varepsilon,$$

and then sets a zero mean[2] GP prior on $f(t)$ and a Gaussian prior on ε:

$$f(t) \sim \mathcal{GP}(0, k(t,t')), \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

where $k(t,t')$ is a covariance function and σ^2 is a hyperparameter that specifies the noise power. The covariance function plays an analogous role to the *kernel* function used in the support vector machine literature (see for instance [20]).

The covariance function $k(t,t')$ specifies the degree of coupling between $y(t)$ and $y(t')$ and encodes properties of the GP such as power level, smoothness, etc. One of the best-known covariance functions, and the one that we will be using here, is the isotropic squared exponential. It has the form of an unnormalized Gaussian,

$$k(t,t') = \sigma_0^2 \exp\left(\frac{-|t-t'|^2}{2\ell^2}\right),$$

and depends on two hyperparameters, σ_0^2 and ℓ, which are collectively referred to as the covariance hyperparameters $\boldsymbol{\theta}$. The power of the GP is controlled by σ_0^2, whereas the smoothness can be selected by tuning the length-scale ℓ. Smaller values for ℓ result in faster decays and therefore correspond to rapidly varying latent functions, whereas a bigger values encode smoother latent functions.

Due to the very definition of GP, the joint distribution of the available observations (collected in $\mathbf{y}$) and some unknown output $y(t_*)$ form a joint multivariate distribution, with parameters specified by the covariance function:

$$\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K_{ff}} + \sigma^2\mathbf{I}_n & \mathbf{k_{f*}} \\ \mathbf{k}_{\mathbf{f}*}^\top & k_{**} + \sigma^2 \end{bmatrix}\right) \quad (1)$$

where $\mathbf{K_{ff}}$, $\mathbf{k_{f*}}$ and k_{**} are a matrix with elements $k(t_i,t_j)$, a vector with elements $k(t_i,t_*)$, and $k(t_*,t_*)$, respectively. $\mathbf{I}_n$ is used to denote the identity matrix of size n.

From (1) and conditioning on the observed training outputs we obtain the desired predictive distribution

$$p_{\mathrm{GP}}(y_*|t_*,\mathcal{D}) = \mathcal{N}(y_*|\mu_{\mathrm{GP}*}, \sigma^2_{\mathrm{GP}*}) \quad (2a)$$

$$\mu_{\mathrm{GP}*} = \mathbf{k}_{\mathbf{f}*}^\top(\mathbf{K_{ff}} + \sigma^2\mathbf{I}_n)^{-1}\mathbf{y} \quad (2b)$$

$$\sigma^2_{\mathrm{GP}*} = \sigma^2 + k_{**} - \mathbf{k}_{\mathbf{f}*}^\top(\mathbf{K_{ff}} + \sigma^2\mathbf{I}_n)^{-1}\mathbf{k}_{\vec{f}*} \quad (2c)$$

which is computable in $\mathcal{O}(n^3)$ time. This cost arises from the inversion of the $n \times n$ matrix $\mathbf{K_{ff}} + \sigma^2\mathbf{I}_n$.

Hyperparameters $\{\boldsymbol{\theta}, \sigma\}$ are typically selected by Type-II Maximum Likelihood, i.e., to maximize the marginal log-likelihood (also called log-evidence) of the observations:

$$\log p(\mathbf{y}|\boldsymbol{\theta},\sigma) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}|\mathbf{K_{ff}} + \sigma^2\mathbf{I}_n| - \frac{1}{2}\mathbf{y}^\top\left(\mathbf{K_{ff}} + \sigma^2\mathbf{I}_n\right)^{-1}\mathbf{y}. \quad (3)$$

[2]It is customary to subtract the sample mean to data $\{y(t_i)\}_{i=1}^n$, and then to assume a zero mean model.

Applicable search algorithms are described in standard textbooks such as [14], [21]. When the analytical derivatives of (3) are available and conjugated gradient ascent is used for optimization, each step takes $\mathcal{O}(n^3)$ time.

If several independent GPs are present (for instance, to model K trajectories over D dimensions), their log-likelihood is simply the sum of the log-likelihoods for each GP:

$$\log p(\mathbf{y}|\boldsymbol{\theta},\sigma) = \sum_{k=1,d=1}^{K,D} -\frac{n_k}{2}\log(2\pi) - \frac{1}{2}|\mathbf{K}_{\mathbf{f_d^k f_d^k}} + \sigma^2\mathbf{I}_{n_k}| - \frac{1}{2}\mathbf{y_{n_k}}^\top\left(\mathbf{K}_{\mathbf{f_d^k f_d^k}} + \sigma^2\mathbf{I}_{n_k}\right)^{-1}\mathbf{y_{n_k}}, \quad (4)$$

where k is used to index the quantities related to each trajectory.

IV. A GP model for the data association problem

In this section we will introduce a Bayesian model for the data association problem based on GPs. Recall that our aim is to infer both the unknown source labels $\mathbf{s}$ and the latent $\{f_d^k(t_i)\}_{d=1,k=1}^{D,K}$.

We proceed in a Bayesian way by placing priors on the unknown parameters of our model. If we assume that trajectories $\{f_d^k(t_i)\}_{d=1,k=1}^{D,K}$ are smooth, it is reasonable to model them with independent GP priors (using some parameters $\boldsymbol{\theta}$ to select the levels of amplitude and smoothness). Noise can be modeled with a Gaussian prior of selectable power σ^2. Since all possible source allocations $\mathbf{s}$ are equally likely a priori, we will place a uniform prior over them. The resulting model can then be described by the likelihood

$$p(y_d(t_i)|\{f_d^k(t_i)\}_{k=1}^K, s_i) = \mathcal{N}(y_d(t_i) \mid f_d^{s_i}(t_i), \sigma^2) \quad (5)$$

and the independent priors

$$p(f_d^k(t_i)) = \mathcal{GP}(0, k(t_i,t_i')) \quad \text{and} \quad P(\mathbf{s}) = 1/K^n \quad (6)$$

(note the use of a uniform, non-informative prior on $\mathbf{s}$).

For notational simplicity, in the following we omit explicit conditioning on hyperparameters $\boldsymbol{\theta}$ and σ^2, which will be regarded as known for the moment, as well as on time instants $\{t_i\}_{i=1}^n$ (which are always known, since they are part of the observations).

A. Label recovery

The posterior probability of any particular set of labels $\mathbf{s}$ (i.e., any possible grouping of data points in trajectories) given the data is obtained from Bayes' formula:

$$P(\mathbf{s}|\mathbf{Y}) = \frac{p(\mathbf{Y}|\mathbf{s})P(\mathbf{s})}{\sum_{\mathbf{s}} p(\mathbf{Y}|\mathbf{s})P(\mathbf{s})} = \frac{p(\mathbf{Y}|\mathbf{s})}{\sum_{\mathbf{s}} p(\mathbf{Y}|\mathbf{s})} \propto p(\mathbf{Y}|\mathbf{s}) \quad (7)$$

where $\mathbf{Y} = \{y_d(t_i)\}_{d=1,i=1}^{D,n}$ collects all available observations.

The first identity of (7) follows because $P(\mathbf{s}) = 1/K^n$ is a constant, uniform prior independent of $\mathbf{s}$. This makes the posterior proportional to the likelihood. The proportionality constant (the denominator) is known, but implies a summation over all K^n possible labellings. As a result, computing it in reasonable time is only possible for very small problems.

The likelihood of $\mathbf{s}$, and therefore its posterior (up to a constant) can be computed in closed form:

$$\log P(\mathbf{s}|\mathbf{Y}) \overset{c}{=} \log p(\mathbf{Y}|\mathbf{s}) = \sum_{k,d} \log p(\mathbf{y}_d^k) = -\frac{nD}{2}\log(2\pi)$$
$$-\frac{1}{2}\sum_{k,d} |\mathbf{K}_{\mathbf{f}_d^k\mathbf{f}_d^k} + \sigma^2\mathbf{I}_{n_k}| + \mathbf{y}_d^{k\top}\left(\mathbf{K}_{\mathbf{f}_d^k\mathbf{f}_d^k} + \sigma^2\mathbf{I}_{n_k}\right)^{-1}\mathbf{y}_d^k \tag{8}$$

where $\mathbf{y}_d^k$ is a column vector collecting all $y_d(t_i)$ such that $s_i = k$ and similarly, $\mathbf{f}_d^k$ is a column vector collecting the latent values $f_d^k(t_i)$ such that $s_i = k$. Without loss of generality, we assume that observations and latent values are ordered within their corresponding vectors according to time. The first equality follows from the independence assumption between different sources and dimensions, introduced in prior (6) in the form of independence[3] among trajectories $\{f_d^k(t_i)\}_{d=1,k=1}^{D,K}$. The second equality corresponds to the evidence of a standard GP (3), independently evaluated at the groups of points specified by $\mathbf{s}$, as described in (4).

The most probable set of labels $\mathbf{s}^*$ (i.e., the MAP estimation) corresponds in this case to the ML estimation, since we are using a flat prior over $\mathbf{s}$:

$$\mathbf{s}^* = \underset{\mathbf{s}}{\operatorname{argmax}}\ \log P(\mathbf{s}|\mathbf{Y}) = \underset{\mathbf{s}}{\operatorname{argmax}}\ \log p(\mathbf{Y}|\mathbf{s}) \tag{9}$$

Since $\mathbf{s}$ is a discrete variable, we can find the mode of the posterior by an exhaustive search over all K^n possible configurations. However, this approach is only valid for very small problems. In Section V we will see how by imposing some additional constraints, it is possible to (approximately) find the mode of the distribution very efficiently.

B. Inferring the trajectories

Once each data point has been allocated to a trajectory via $\mathbf{s}$, standard GP regression can be applied to infer the latent functions describing the trajectories. Fully probabilistic predictions for the value of any observation at new time instant t_* can be obtained as follows

$$p(y_d^k(t_*)|\mathbf{Y},\mathbf{s}) = \mathcal{N}(y_d^k(t_*)|\mu_*, \sigma_*^2) \tag{10a}$$
$$\mu_* = \mathbf{k}_{\mathbf{f}_k^d *}^\top(\mathbf{K}_{\mathbf{f}_k^d\mathbf{f}_k^d} + \sigma^2\mathbf{I}_{n_k})^{-1}\mathbf{y} \tag{10b}$$
$$\sigma_*^2 = \sigma^2 + k_{**} - \mathbf{k}_{\mathbf{f}_k^d *}^\top(\mathbf{K}_{\mathbf{f}_k^d\mathbf{f}_k^d} + \sigma^2\mathbf{I}_{n_k})^{-1}\mathbf{k}_{\mathbf{f}_k^d *}. \tag{10c}$$

This is a standard result from the GP literature, see for instance [14].

A fully Bayesian method should integrate over all possible configurations $\mathbf{s}$ to obtain the predictive distribution $p(y_d^k(t_*)|\mathbf{Y})$, but again this would require a summation over K^n values. It is possible to obtain a reasonably good estimate by considering only the most probable configuration

$$p(y_d^k(t_*)|\mathbf{Y}) = \sum_{\mathbf{s}} p(y_d^k(t_*)|\mathcal{D},\mathbf{s})P(\mathbf{s}|\mathbf{Y}) \approx p(y_d^k(t_*)|\mathbf{Y},\mathbf{s}^*),$$

i.e., replacing $\mathbf{s}$ with $\mathbf{s}^*$ in (10).

[3]Introducing dependencies among different sources or dimensions does not prevent tractability, and can be simply achieved by modifying the prior. It increases computational complexity, though.

C. Model selection

We have seen that we select the best configuration $\mathbf{s}$ according to the ML criterion. For GPs, it is standard practice to select hyperparameters using ML, so we can express both the model selection and the best configuration selection as a joint maximization:

$$\{\mathbf{s}^*,\ \boldsymbol{\theta},\ \sigma^2\} = \underset{\mathbf{s}^*,\boldsymbol{\theta},\sigma^2}{\operatorname{argmin}}\ -\log p(\mathbf{Y}|\mathbf{s},\boldsymbol{\theta},\sigma^2) \tag{11}$$

where we have explicitly included the conditioning on $\boldsymbol{\theta}$ and σ^2. Cost functional $-\log p(\mathbf{Y}|\mathbf{s},\boldsymbol{\theta},\sigma^2)$ could even be used to select the most likely number of sources K of the model: Increasing the number of the clusters only affects the independence relations appearing in the prior, but does not necessarily reduce the cost, unlike other clustering models.

V. Reduction to a MaxCut problem

In this section we will introduce a simplifying assumption (which often holds in practice) that converts optimization problem (8) into a constrained Max K-Section problem. Then we will consider the $K = 2$ case, which further simplifies the objective, yielding an unconstrained MaxCut problem. Finally, we provide an efficient algorithm to address the data association problem in the mentioned $K = 2$ case, in which only two trajectories are involved.

A. The data association problem as a constrained Max $K-$Section problem

We will first introduce a simplifying constraint in the previous model: We will assume that *all sources* produce exactly one vector observation $[y_1(t_i),\dots,y_D(t_i)]^\top$ at *every time instant*. Then, $\mathbf{f}_d^k$, which only collects the latent values $f_d^k(t_i)$ such that $s_i = k$, will now collect every latent value exactly once, regardless of k. Thus, under this constraint, we have that $\mathbf{K}_{\mathbf{f}_d^k\mathbf{f}_d^k} = \mathbf{K}_{\mathbf{ff}}$ (i.e., the covariance matrix corresponding to each source and dimension no longer depends on the allocation $\mathbf{s}$). Substituting in (8) and removing additive terms that do not depend on $\mathbf{s}$, we have:

$$\log P(\mathbf{s}|\mathbf{Y}) \overset{c}{=} -\frac{1}{2}\sum_{k,d}\mathbf{y}_d^{k\top}\left(\mathbf{K}_{\mathbf{ff}} + \sigma^2\mathbf{I}_{n'}\right)^{-1}\mathbf{y}_d^k. \tag{12}$$

For any given dimension d, each vector $\mathbf{y}_d^k$ collects the $n' = n/K$ observations flagged by $\mathbf{s}$ as belonging to trajectory k. Therefore, the dependence of (12) on $\mathbf{s}$ is expressed by how each actual observation is allocated to each $\mathbf{y}_d^k$. This obscures the dependence of $\log P(\mathbf{s}|\mathbf{Y})$ on $\mathbf{s}$, which does not appear explicitly in the expression.

In order to clarify this dependence, we need the following two definitions: First, we collect all available observations for dimension d in diagonal[4] matrix $\mathbf{A}_d = \operatorname{diag}([\mathbf{y}_d^{1\top},\ \dots,\ \mathbf{y}_d^{K\top}]$, of size $n \times n$, *using any arbitrary valid allocation*. I.e., for each time instant t_i, we randomly assign each of the K available observations to one of the K available sources and build matrices $\{\mathbf{A}_d\}_{d=1}^D$ according to that allocation.

[4]We use operator $\operatorname{diag}(\cdot)$ to convert a vector into matrix by arranging the elements of the former in the main diagonal of the latter.

Second, we express $\mathbf{s}$ as an equivalent set of K binary indicator vectors $\{\mathbf{b}^k\}_{k=1}^K$, each of length n, consisting only of zeros and ones. Each element of indicator $\mathbf{b}^k$ is set to 1 if the corresponding element of the diagonal of matrix $\{\mathbf{A}_d\}_{d=1}^D$ belongs to source k and to 0 otherwise. Note that this makes the choice of the arbitrary allocation used to create $\{\mathbf{A}_d\}_{d=1}^D$ irrelevant: Any allocation $\mathbf{s}$ can be expressed by a corresponding set of $\{\mathbf{b}^k\}_{k=1}^K$.

Using these definitions, we can rearrange (12) as:

$$\begin{aligned}\log P(\mathbf{s}|\mathbf{Y}) &\overset{c}{=} \log P(\mathbf{b}|\mathbf{Y}) \overset{c}{=} \\ &-\frac{1}{2}\sum_{k,d}\mathbf{b}^{k\top}\mathbf{A}_d[\left(\mathbf{K}_{\mathbf{ff}}+\sigma^2\mathbf{I}_{n'}\right)^{-1}\otimes(\mathbf{1}_K\mathbf{1}_K^\top)]\mathbf{A}_d\mathbf{b}^k \\ &= -\frac{1}{2}\sum_{k=1}^K\mathbf{b}^{k\top}\mathbf{M}\mathbf{b}^k \end{aligned} \tag{13}$$

where $\mathbf{M}=\sum_{d=1}^D\mathbf{A}_d[\left(\mathbf{K}_{\mathbf{ff}}+\sigma^2\mathbf{I}_{n'}\right)^{-1}\otimes(\mathbf{1}_K\mathbf{1}_K^\top)]\mathbf{A}_d$, the tensor product operator is $\otimes$ and $\mathbf{1}_K$ is a column vector with K ones.

Eq. (13) clearly elucidates the dependence of the posterior probability on $\mathbf{b}$. Note that matrix $\mathbf{M}$ is constant (for given hyperparameters $\boldsymbol{\theta}$ and σ). If we include the constraints that $\{\mathbf{b}^k\}_{k=1}^K$ must fulfill to represent a valid allocation, we have the following optimization problem:

$$\mathbf{b}^* = \underset{\{\mathbf{b}^k\}_{k=1}^K}{\operatorname{argmin}}\ \frac{1}{2}\sum_{k=1}^K\mathbf{b}^{k\top}\mathbf{M}\mathbf{b}^k \tag{14a}$$

$$\text{s.t.}\quad \mathbf{b}^k\in\{0,1\}^n \tag{14b}$$

$$\sum_{k=1}^K\mathbf{b}^k=\mathbf{1}_n \tag{14c}$$

$$\sum_{i=1}^n[\mathbf{b}^k]_i=\mathbf{1}_n^\top\mathbf{b}^k=\frac{n}{K}=n',\forall_{k=1,\dots,K} \tag{14d}$$

$$\sum_{j=0}^{K-1}[\mathbf{b}^k]_{i+jn'}=1,\forall_{k=1,\dots,K},\forall_{i=1,\dots,n'} \tag{14e}$$

The minimization objective can be alternatively expressed in terms of positive weights as $\frac{1}{2}\sum_{k=1}^K\mathbf{b}^{k\top}\mathbf{W}\mathbf{b}^k$, with $\mathbf{W}=\mathbf{M}+w\mathbf{1}_n\mathbf{1}_n^\top$ and w an arbitrary constant, big enough to make all the elements of $\mathbf{W}$ positive. Due to constraint (14d), this modification only shifts the objective by a constant, and therefore does not modify the optimal $\mathbf{b}^*$:

$$\begin{aligned}\frac{1}{2}\sum_{k=1}^K\mathbf{b}^{k\top}\mathbf{W}\mathbf{b}^k &= \frac{1}{2}\sum_{k=1}^K\mathbf{b}^{k\top}(\mathbf{M}+w\mathbf{1}_n\mathbf{1}_n^\top)\mathbf{b}^k \\ &= \frac{1}{2}\sum_{k=1}^K\mathbf{b}^{k\top}\mathbf{M}\mathbf{b}^k+\frac{w}{2}\sum_{k=1}^K\mathbf{b}^{k\top}\mathbf{1}_n\mathbf{1}_n^\top\mathbf{b}^k \\ &= \frac{1}{2}\sum_{k=1}^K\mathbf{b}^{k\top}\mathbf{M}\mathbf{b}^k+\frac{wKn'^2}{2}.\end{aligned}$$

If the binary indicator $\mathbf{b}$ is only constrained by (14c), which enforces *mutual exclusiveness* (i.e., each observation must come from a single source), the minimization of $\frac{1}{2}\sum_{k=1}^K\mathbf{b}^{k\top}\mathbf{W}\mathbf{b}^k$ is a standard Max K-cut problem, which can be approximately solved by relaxing it to an SDP problem as described in [22], [23]. Observe that this formulation expresses the Max K-cut problem as the desire of minimizing the weight of intra-partition edges, whereas an equivalent, more standard formulation would require maximizing the weight of inter-partition edges.

The inclusion of the *balance* constraint (14d), which forces to allocate the same number of points to every source (partition), yields the Max K-Section problem. Just as Max K-cut, it can be solved using an SDP relaxation, as detailed in [24].

Finally, the *non-simultaneity* constraint (14e) avoids assigning two points occurring in the same time instant to the same source (partition). We conjecture that this additional linear constraint still results in a problem that can be solved using an SDP relaxation, probably a simple modification of the algorithm proposed in [24]. We are not currently aware of any work addressing this constrained Max K-Section problem and leave it as an open problem that will be the subject of further research.

B. *The $K = 2$ case and MaxCut*

If we know that only two trajectories are present in our observations, then $K=2$, and the previous problem can be considerably simplified.

In this case, at each of the $\frac{n}{2}$ time instants we have two generating sources and two observations, and we must decide which source generated which observation. Since there are obviously only two different ways in which this observations could have been generated, we can code all possible allocations for all time instants ($\mathbf{s}$) using a binary vector $\mathbf{h}\in[-1,+1]^{n/2}$.

Following the notation described in Section V-A, the binary vectors $\mathbf{b}^1$ and $\mathbf{b}^2$ can be obtained as a function of $\mathbf{h}$. Since there are only two partitions, indicators $\mathbf{b}^1$ and $\mathbf{b}^2$ are complementary, i.e. if element $[\mathbf{b}^1]_i$ is 1, $[\mathbf{b}^2]_i$ must be 0 and vice versa. Furthermore, because $[\mathbf{b}^k]_i$ corresponds to the same time instant as $[\mathbf{b}^k]_{i+\frac{n}{2}}$, one of them must be 1 and the other 0. Thus we have:

$$\mathbf{b}^1=\frac{1}{2}\left(\mathbf{1}_n+\begin{bmatrix}+\mathbf{h}\\-\mathbf{h}\end{bmatrix}\right),\qquad \mathbf{b}^2=\frac{1}{2}\left(\mathbf{1}_n-\begin{bmatrix}+\mathbf{h}\\-\mathbf{h}\end{bmatrix}\right). \tag{15}$$

Plugging (15) in (14), we get the following simplified minimization problem:

$$\mathbf{h}^*=\underset{\mathbf{h}}{\operatorname{argmin}}\ \frac{1}{2}\mathbf{h}^\top\mathbf{Q}\mathbf{h}+\frac{1}{2}\mathbf{1}_n\mathbf{M}\mathbf{1}_n \tag{16a}$$

$$\text{s.t.}\quad \mathbf{h}^k\in\{-1,1\}^n \tag{16b}$$

with $\mathbf{Q}=\sum_1^D\operatorname{diag}(\mathbf{y}_d^1-\mathbf{y}_d^2)\left(\mathbf{K}_{\mathbf{ff}}+\sigma^2\mathbf{I}_{n'}\right)^{-1}\operatorname{diag}(\mathbf{y}_d^1-\mathbf{y}_d^2)$.

Observe how all the constraints in (14) are automatically fulfilled due to the way in which $\mathbf{b}^1$ and $\mathbf{b}^2$ are constructed from $\mathbf{h}$. Any $\mathbf{h}$ produces $\mathbf{b}^k$, fulfilling mutual exclusiveness, balance and non-simultaneity constraints, so there is no need to place constraints on $\mathbf{h}$.

Excluding the term $\frac{1}{2}\mathbf{1}_n\mathbf{M}\mathbf{1}_n$, which is constant and therefore irrelevant in the minimization, (16) describes a standard MaxCut problem. Matrix $\mathbf{Q}$ is positive semidefinite, though its elements are not necessarily positive. The optimal label allocation $\mathbf{h}^*$ can then be found using an SDP relaxation.

C. Solution using an SDP relaxation

To obtain a (remarkably good) approximation to the solution of problem (16), we proceed as described in [25].

First, we define $\mathbf{H} = \mathbf{h}\mathbf{h}^\top$ and present optimization problem (16) in the following equivalent form:

$$\mathbf{H}^* = \underset{\mathbf{h}}{\operatorname{argmin}} \operatorname{trace}(\mathbf{HQ}) \tag{17a}$$

$$\text{s.t.} \quad \mathbf{H} \succeq 0, \quad \operatorname{diag}(\mathbf{H}) = \mathbf{1}_{\frac{n}{2}} \tag{17b}$$

$$\operatorname{rank}(\mathbf{H}) = 1 \tag{17c}$$

Then, we relax the optimization problem (17), which is exactly equivalent to (16), by dropping constraint (17c), so that it becomes convex and can be efficiently solved using SDP techniques.

The (approximately) optimal labeling $\mathbf{h}^*$ is retrieved from $\mathbf{H}^*$ using random hyperplane rounding as follows: Use the Cholesky decomposition to expand $\mathbf{H}^* = \mathbf{V}^\top\mathbf{V}$ and associate each column from $\mathbf{V}$ (which, due to constraint (17b) is a vector located in the surface of a unitary hypersphere) with the element from $\mathbf{h}^*$ in the same position. Then generate a random hyperplane through the origin and assign +1 to the elements of $\mathbf{h}^*$ corresponding to columns of $\mathbf{V}$ lying on one side of the hyperplane and -1 to the elements corresponding to columns lying the other side.

It can be proved that this procedure yields an 0.87856-approximation to the optimal solution (see [25]). Since cost function (16) can be evaluated very efficiently for any candidate solution, this process can be repeated several times with different random hyperplanes and the best solution chosen.

D. A worked example

To clarify the ideas developed in this section, let us work through a simple example. Assume that at time instant $t_1 = -2$ we have two output observations, -2 and 2. Likewise, at $t_2 = -1$ we observe 0 and 1; and at $t_2 = 1$ we observe -2 and 4.5 (see Fig. 2). Note that at each time instant we have exactly two observations, each from a different source, but it is not known which source generates which observation. There are $2^{\frac{n}{2}}$ (in this case, 2^3) possible ways to allocate observations to sources. For each possible allocation $\mathbf{s}$, we can collect observations labeled as coming from source 1 in $\mathbf{y}^1$ and observations labeled as coming from source 2 in $\mathbf{y}^2$. Then we can compute $-\log P(\mathbf{s}|\mathbf{Y}) \overset{c}{=} \sum_{k\in\{1,2\}} \mathbf{y}^k(\mathbf{K_{ff}} + \sigma^2\mathbf{I}_3)^{-1}\mathbf{y}^k$ as per eq. (8). Note that optimizing this expression w.r.t. to $\mathbf{s}$ to compute the MAP solution for the labeling involves recomputing it $2^{\frac{n}{2}}$ times, one for each possible allocation, and then selecting the one that yields the maximum value. The combinatorial explosion makes this unaffordable even for medium sized n.

Following the previous derivations of this section, we can just assume any arbitrary allocation to fill $\mathbf{y}^1$ and $\mathbf{y}^2$ and write $-\log P(\mathbf{s}|\mathbf{Y})$ (up to a constant) as

$$\frac{1}{2}\mathbf{h}^\top \operatorname{diag}(\mathbf{y}^1 - \mathbf{y}^2)\left(\mathbf{K_{ff}} + \sigma^2\mathbf{I}_3\right)^{-1}\operatorname{diag}(\mathbf{y}^1 - \mathbf{y}^2)\mathbf{h}$$
$$= \frac{1}{2}\mathbf{h}^\top \operatorname{diag}([4, 1, 6.5])\left(\mathbf{K_{ff}} + \sigma^2\mathbf{I}_3\right)^{-1}\operatorname{diag}([4, 1, 6.5])\mathbf{h}$$

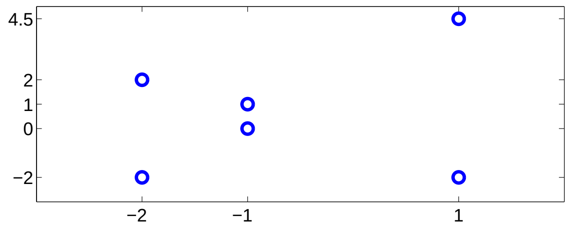


Fig. 2. Data for the worked example.

with $\mathbf{h} \in \{-1, +1\}^3$ as per eq. (16). Each element of vector $\mathbf{h}$ tells us, for each time instant, if our initial arbitrary allocation is kept $(+1)$ or reversed (-1) in the computation of $-\log P(\mathbf{s}|\mathbf{Y})$. Thus, optimizing over $\mathbf{h}$ yields the optimal allocation. In order to solve this problem, we still need to compute this expression for the $2^{\frac{n}{2}}$ possible values of $\mathbf{h}$, which is unacceptable.

If we now assume for the sake of simplicity that we use a linear kernel $k(t,' t) = tt'$ and unit noise power $\sigma^2 = 1$, we can instead minimize the relaxed problem (17) $\operatorname{trace}(\mathbf{HQ})$ s.t. $\mathbf{H} \succeq 0, \operatorname{diag}(\mathbf{H}) = \mathbf{1}_3$ to get $\mathbf{H}^*$, with

$$\mathbf{Q} = \begin{bmatrix} +.31 & +.50 & -.08 \\ +.50 & +2.0 & -.15 \\ -.08 & -.15 & +.05 \end{bmatrix}^{-1}, \mathbf{H}^* = \begin{bmatrix} +1 & +1 & -1 \\ +1 & +1 & -1 \\ -1 & -1 & +1 \end{bmatrix}$$

Using the hyperplane rounding technique, from $\mathbf{H}^*$ we get $\mathbf{h} = [+1, +1, -1]$, thus indicating that found trajectories are $[-2, 0, 4.5]$ and $[2, 1, -2]$, each from a different source.

VI. The GP-BTT algorithm (including hyperparameter learning)

For the case in which only two trajectories are present, it is possible to learn the hyperparameters and infer the labeling using an efficient algorithm, which we describe below.

Ideally, we would like to minimize (11) w.r.t. both hyperparameters $\{\boldsymbol{\theta}, \sigma^2\}$ and the allocation (represented by $\mathbf{s}$ in the general case, and more compactly expressed by $\mathbf{h}$ in the binary case). Instead, we have shown so far how to carry out this minimization for each case: If the allocation was known, it would be possible to minimize (11) w.r.t. the hyperparameters by using simple conjugate gradient descent, which is standard practice when learning regular GPs. If the hyperparameters were known, it would possible to approximately compute the optimal allocation $\mathbf{h}^*$, casting the problem as a MaxCut algorithm, as per Section (V-B), and using the SDP relaxation described in Section (V-C).

To combine both types of optimization, several strategies can be used. A natural and straightforward option would be to iteratively alternate between optimizing the allocation and learning the hyperparameters, thus giving rise to an EM-like algorithm. However, if this technique is directly applied, the quality of the obtained local minima will depend strongly on the initialization. Instead, we propose to use a few different initializations and then optimize the allocations and learn the hyperparameters only once. This scheme amounts to a joint search as the number of initializations grows, but in practice produces very good results when just a few are used. Of course, other optimization schemes are possible.

In detail, the resulting algorithm, which we call GP Binary-Target Tracking (GP-BTT) is:

1) Input: We are given a set of time instants $\{t_i\}_{i=1}^{n/2}$ and two corresponding vectors of observations $\mathbf{y}_d^1$ and $\mathbf{y}_d^2$ per dimension, each of size $\frac{n}{2}$. (The two observations available at each time instant i are randomly assigned to $\{[\mathbf{y}_d^1]_i\}_{d=1}^D$ and $\{[\mathbf{y}_d^2]_i\}_{d=1}^D$; the goal of the algorithm is to determine which source produced which observation).
2) Form a 2-dimensional grid of candidate values for ℓ and the Signal-to-noise ratio SNR $= \sigma_0^2/\sigma^2$, covering a sufficiently wide range.
3) For each candidate pair $\{\ell, \text{SNR}\}$
    a) Set σ_0^2 to the variance of data[5], compute $\sigma^2 = \sigma_0^2/\text{SNR}$ and build $\mathbf{Q}$ from data.
    b) Compute $\mathbf{h}^*$ using the MaxCut algorithm described in Section (V-C).
    c) Reorder the observation vectors to reflect the obtained clustering. For every i:
        - If $[\mathbf{h}^*]_i = +1$, do nothing.
        - If $[\mathbf{h}^*]_i = -1$, swap $[\mathbf{y}_d^1]_i$ and $[\mathbf{y}_d^1]_i$, for every d.
    d) Now $\{\mathbf{y}_d^1\}_{d=1}^D$ and $\{\mathbf{y}_d^1\}_{d=1}^D$ constitute $2D$ independent GPs. Learn their hyperparameters maximizing (4), using a gradient based procedure (such as conjugate gradient), starting from their current values.
    e) Compute the negative log marginal likelihood (NLML) for the selected hyperparameters, with Eq. (4). If its better than the NLML of previous iterations, store the final values of the hyperparameters $\{\sigma_0^2, \sigma^2, \ell\}$, the allocations $\mathbf{h}^*$ and the resulting NLML obtained for this candidate pair.
4) Output the allocation $\mathbf{h}^*$ and hyperparameters corresponding to the best NLML.

In practice, the algorithm seems fairly robust to the choice or the SNR, and to a lesser extent, to the choice of the length-scale. Depending on the computational requirements, the search grid can be very small, and even reasonable fixed values for both parameters can be used, without incurring in a dramatic reduction in the clustering quality.

When data for $n/2$ different time instants is provided, the described method requires $\mathcal{O}(n^2)$ space (dominated by the storage of the $n/2 \times n/2$ covariance matrix) and $\mathcal{O}(n^3)$ time (dominated by the SDP optimization process, see Fig. 4).

VII. Experiments

In this section we apply the GP-BTT algorithm to several problems with $K = 2$. We also compare is results to different state-of-the-art data association algorithms including the SIR/MCJPDA filter [8], ClusterTrack [3] and OMGP [13].

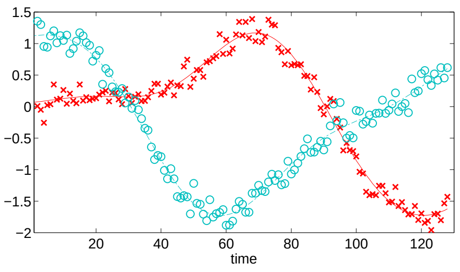


Fig. 3. GP-BTT solution for the toy data set of Fig. 1.

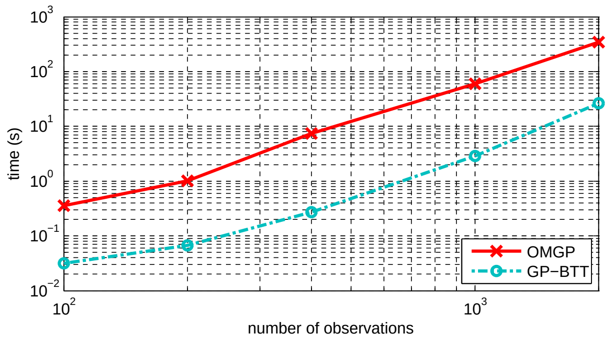


Fig. 4. Comparison of running times of OMGP versus GP-BTT.

A. Toy data

We first apply the GP-BTT algorithm on the data of Fig. 1, which represents the noisy one-dimensional observations of two particles that cross twice. The tracking results of GP-BTT are shown in Fig. 3, in which circles and crosses indicate the obtained clustering solution and hence the estimated data association, and full lines mark the inferred trajectories. GP-BTT considers all possible data association solutions in order to recovers the data associations, but by relaxing the problem to an SDP problem it avoids the combinatorial explosion associated to evaluating all of them explicitly.

In order to get an idea of the time complexity of GP-BTT, we show its running times for different numbers of observations in Fig. 4, obtained on a Pentium Intel Core2 Duo machine with 3GHz processors running Matlab 7.11. As can be seen, GP-BTT is remarkably faster than OMGP, typically a factor of 10 or greater.

B. Missile-to-air tracking scenario

Next, we consider a missile-to-air tracking scenario as described in [8]. This scenario follows a state-space model, in which the state vector contains the position and velocity components of a source, $\mathbf{s}_t = [X_t, Y_t, Z_t, V_{x,t}, V_{y,t}, V_{z,t}]$. The full motion dynamics are defined by the following state-space

[5]Note that for a given SNR and ℓ, the value of σ_0^2 is irrelevant, since it only would only scale $\mathbf{K_{ff}}$, and the solution of (16), $\mathbf{h}^*$, is invariant to scalings of $\mathbf{K_{ff}}$.

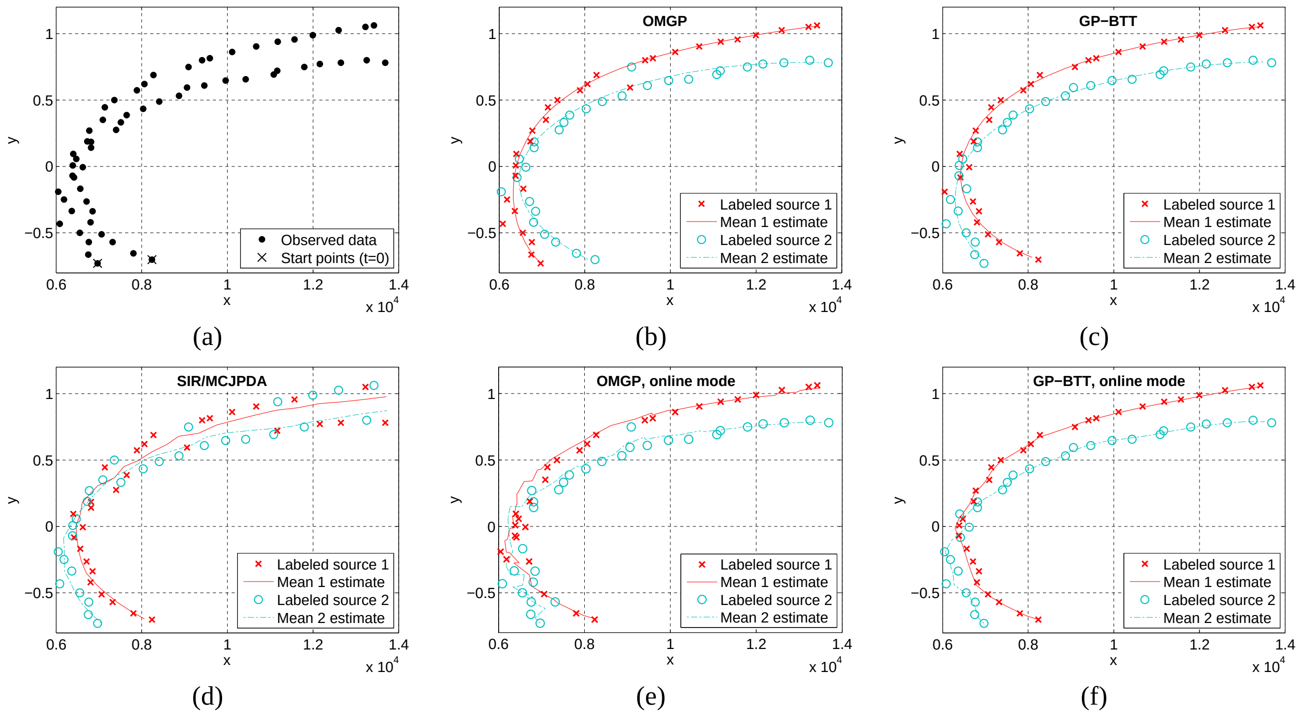


Fig. 5. Data and results for the missile-to-air tracking scenario. The z and time axis are not displayed. Top row: Observed data (a). (b) Data association results of OMGP. (c) Data association results of GP-BTT. (d) Tracking results of the SIR/MCJPDA online filter. (e) Tracking results of OMGP in online mode. (f) Tracking results of GP-BTT in online mode.

equations:

$$\mathbf{s}_{t+1} = \begin{bmatrix} \mathbf{I}_3 & T\mathbf{I}_3 \\ \mathbf{O}_3 & \mathbf{I}_3 \end{bmatrix} \mathbf{s}_t + \begin{bmatrix} \frac{T^2}{2}\mathbf{I}_3 \\ T\mathbf{I}_3 \end{bmatrix} \mathbf{v}_t,$$
$$\mathbf{r}_t = h(\mathbf{s}_t) = \begin{bmatrix} \sqrt{X_t^2+Y_t^2+Z_t^2} \\ \arctan\left(\frac{Y_t}{X_t}\right) \\ \arctan\left(\frac{-Z_t}{\sqrt{X_t^2+Y_t^2}}\right) \end{bmatrix} + \mathbf{e}_t, \quad (18)$$

where T is the sampling interval, and $\mathbf{I}_3$ and $\mathbf{O}_3$ represent the 3×3 unit matrix and null matrix, respectively. The vector $\mathbf{r}_t$ contains the observation at time instant t. The process noise $\mathbf{v}_t$ and measurement noise $\mathbf{e}_t$ are assumed Gaussian, $\mathbf{v}_t \in \mathcal{N}(0,\mathbf{Q})$ with $\mathbf{Q} = \text{diag}(10^2, 10^2, 10^2)$ and $\mathbf{e}_t \in \mathcal{N}(0,\mathbf{R})$ with $\mathbf{R} = \text{diag}(50^2, 0.01^2, 0.01^2)$. For additional details refer to [8]. We consider a bi-target tracking problem with initial states

$$\mathbf{s}_0^1 = [6000, -5000, 2000, 10, 550, 0]^T,$$
$$\mathbf{s}_0^2 = [5050, -4500, 2000, 100, 500, 0]^T. \quad (19)$$

This choice of the initial states results in trajectories that are very close and similar, making this a particularly hard multi-target tracking problem. The corresponding observations $\mathbf{r}_t$ are displayed in Fig. 5(a).

We compare the results of the GP-BTT algorithm with the SIR/MCJPDA filter from [8] and the OMGP algorithm from [13]. The SIR/MCJPDA filter is a state-of-the-art multi-target tracking algorithm that consists of a set of joint particle filters that perform tracking of multiple sources, combined with a joint probability data association (JPDA) technique which provides instantaneous data association. The number of particles used for this experiment is fixed to 100 per source. OMGP is a data association algorithm based on the same mixture model as GP-BTT. OMGP considers a more general scenario in which the number of sources may be higher than 2, but since it requires to solve a nonlinear optimization problem it is less robust to noise than GP-BTT.

In order to operate correctly, the SIR/MCJPDA filter requires complete knowledge of the true dynamics of the model, including the initial states. In contrast, OMGP and GP-BTT are completely blind with regard to the initial states: They only require to determine the hyperparameters, in particular the length-scale l and the data SNR σ_0^2/σ^2. Since this application represents an online scenario in which the running time should be kept low, we did not use the automatic learning of the hyperparameters as described in Section VI. Rather, we fixed the hyperparameters as $\ell = 10$ and SNR $= 100$, based on the amplitudes used in the state-space model. Concerning the robustness of these choices, we verified that changes in the SNR of up to one order of magnitude did not affect the clustering results. The choice of the correct length-scale is more critical, akin to the choice of the kernel width in support vector machine (SVM) literature. Similarly, the SNR is akin to the regularization in SVMs.

SIR/MCJPDA is an online algorithm and thus performs each prediction without knowledge of future data, in contrast to the batch algorithms OMGP and GP-BTT. In order to make a fair comparison, we also run OMGP and GP-BTT in an online manner by performing these batch algorithms on a growing window of observations: The tracking solution at instant t is then only based on the observations up tilt the instant t.

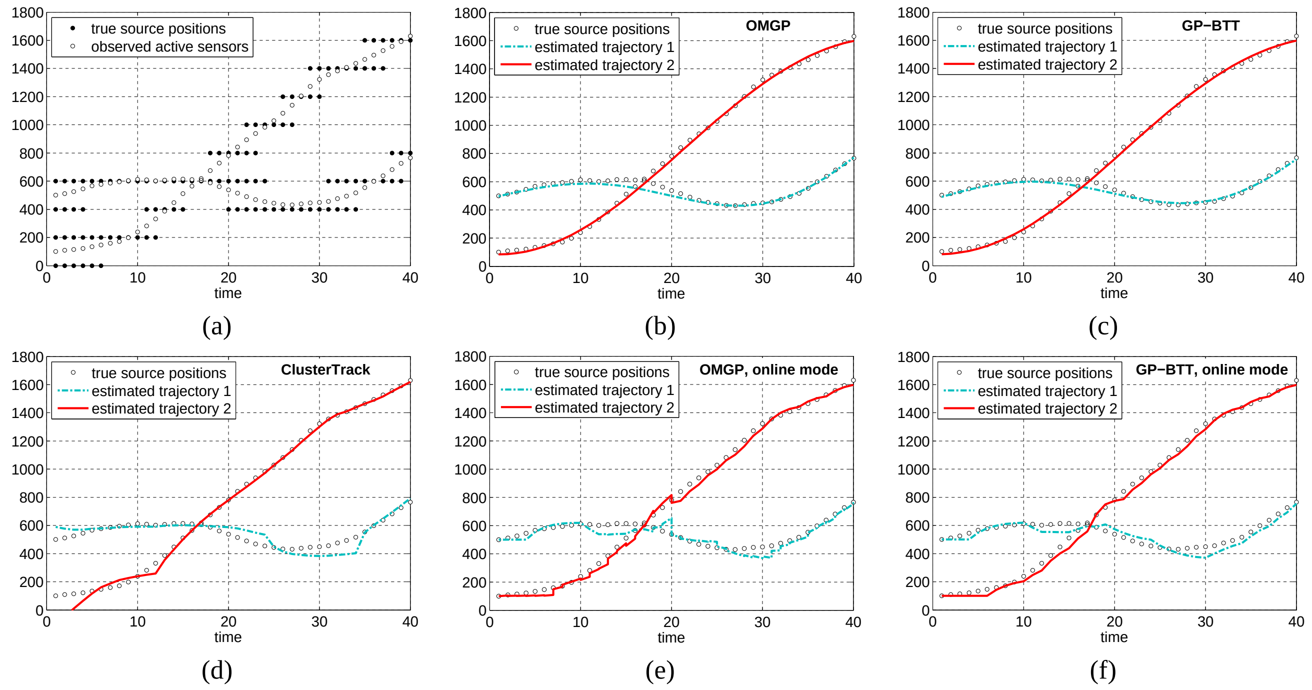


Fig. 6. Data and results for the binary proximity sensor experiment. (a) Observed data. (b) Data association results obtained by OMGP. (b) Data association results obtained by GP-BTT. (d) Tracking results of the ClusterTrack algorithm from [3]. (e) Tracking results of OMGP in online mode. (f) Tracking results of GP-BTT in online mode.

TABLE I
NUMBER OF LABELING ERRORS ON THE MISSILE-TO-AIR DATA ASSOCIATION PROBLEM, OUT OF 60 OBSERVATIONS.

mode	SIR/MCJPDA	OMGP	GP-BTT
batch	n/a	18	4
online	20	11	4

The number of labeling errors obtained by each method are listed in Table I. We also plot the trajectories obtained by the three algorithms in Fig. 5, along with the predicted measurements. Figures 5(b) and (c) illustrate the solutions of OMGP and GP-BTT when applied as batch algorithms on all available data. While OMGP retrieves smooth trajectories, it makes a data association mistake at the instant when both observation are close, causing it to swap labels after this point. GP-BTT does not commit this error.

The result of the SIR/MCJPDA filter, shown in Fig. 5(d), is initially correct for both trajectories, though it becomes erroneous at the instant where the observations are very close. From this point on the results are very poor. Figures 5(e) and (f) show the results of OMGP and GP-BTT in online mode. While the tracking solutions are not as smooth as the batch algorithms other algorithms, their label assignments show few errors. Finally, note that both OMGP and GP-BTT require much less problem-specific information than SIR/MCJPDA to reach their results.

C. *Target tracking with binary proximity sensors*

For the next experiment we consider the problem of target tracking with a grid of binary proximity sensors. This problem has recently drawn a lot of interest as a promising application of wireless sensor networks [26]. In this scenario we consider binary sensors that produce a single bit as their output, which is 1 when one or more targets are in its sensing range R, and 0 otherwise. Despite the minimal information provided by an individual binary proximity sensor, a network of such sensors can provide remarkably good target tracking performance of a single target [26]. The problem becomes more complex if multiple targets are present, and only few methods have dealt with this situation. In particular, a tracking algorithm based on particle filtering was recently proposed in [3], called ClusterTrack. We will conduct an experiment to compare the performance of this algorithm and the proposed GP-BTT method.

We simulate a scenario with two targets that move through a one-dimensional sensor array of 9 sensors. The sensors are positioned uniformly at intervals of $S = 200$ distance units, and their ideal sensing radius is fixed as $R = 0.75S$. These ranges guarantee that a target is always detected by at least one sensor when it is moving through the array. To obtain the target movements, we consider only the x-components of the state-space model (18), with initial states $\mathbf{s}_0^1 = [500, 10]^T$, $\mathbf{s}_0^2 = [100, 10]^T$. Fig. 6(a) shows the observations of this model measured during 40 time instants. The targets cross each other around $t = 17$. Compared to the examples in [3], the number of sensors in this experiment is very low and the targets are very close to each other during the entire experiment, giving rise to a hard tracking problem.

Figure 6(b) shows the data association results obtained by the OMGP algorithm from [13]. In order to apply the GP-BTT algorithm to the data, exactly two measurements per time instant are required. We therefore preprocess the data as follows. If the activated sensors comprise two disjoint groups (i.e. there is at least one deactivated sensor between them), the centre of each group is used as the input to the algorithm. Otherwise, the two centres are chosen as the positions that maximize the likelihood of the observations at that time instant. The results for of the GP-BTT algorithm with hyperparameters $l = 20$ and SNR $= 100$ are shown in 6(c). Despite the minimal information provided by the binary sensors, it can be observed that the batch versions of both OMGP and GP-BTT obtain very reasonable results.

Next, we apply the ClusterTrack algorithm from [3] to these data, with 100 particles per target. ClusterTrack uses complete knowledge of the state-space model in order to operate correctly (see [3] for additional details). While this algorithm is capable of estimating the number of sources by performing multiple runs over the data, we consider its online mode, which has knowledge of the number of sources and only performs one run. Fig. 6(d) shows its tracking results. The source positions are estimated with great precision at most time instants. Nevertheless, it also shows some regions of higher error, particularly around the start, and between $t = 20$ and $t = 35$ for the second trajectory. We then repeat the experiment using the OMGP and GP-BTT algorithms in online mode. As shown in Figs. 6(e) and (f), the online algorithms obtain trajectories that are more irregular compared to the batch algorithms. Still, the estimated trajectories are reasonably close to the true, unobservable source positions. The fact that the results for OMGP are practically identical to those of GP-BTT should not come as a surprise, as OMGP typically performs well in problem with little noise. Note also that OMGP and GP-BTT in online mode performs similar to the ClusterTrack algorithm, which is specifically tailored to handle this problem.

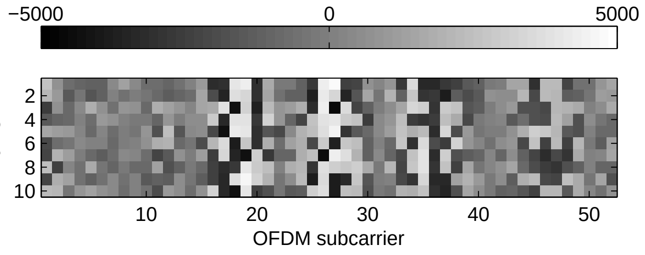
D. Blind decoding of BPSK symbols in OFDM transmissions

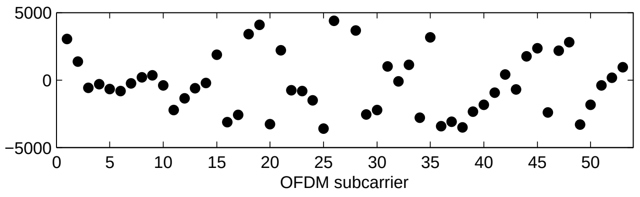
For the fourth experiment we consider a data association problem from the field of wireless communication networks. The setting includes a high level of noise, which allows us to assess the robustness of GP-BTT to noise.

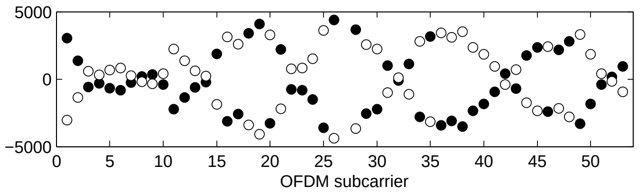
The IEEE Standard 802.11a-1999 describes the data transmission in wireless local area network (WLAN) computer communications [27]. According to these specifications, data is transmitted using orthogonal frequency-division multiplexing (OFDM) modulation with 52 subcarrier frequencies [28]. In other words, each data packet is split up into several sequences and each sequence is transmitted over a different subcarrier frequency, during several time frames. The sequences themselves can be chosen from several different constellations, in particular "binary phase-shift keying" (BPSK), which uses binary symbols $\in \{-1, +1\}$. BPSK is a robust modulation, used typically when noise levels are high.

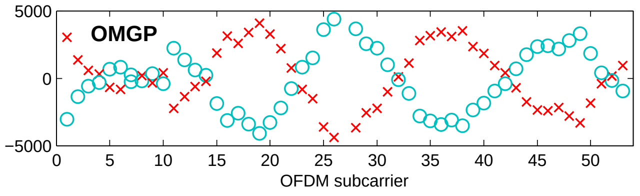
During transmission, the symbols are corrupted by the wireless channel and they have to be recovered at the receiver side. Fig. 7(a) shows the real part[6] of the first 10 time frames of a received data packet. At each time frame, data is received in each of the subcarrier frequencies. The variations in the

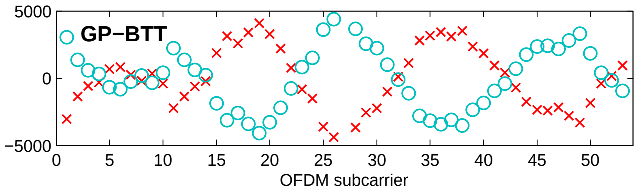
[6]Since the wireless channel has a complex impulse response, the received signal is complex as well.



(a)


(b)


(c)


(d)


(e)

Fig. 7. Data and results of the BPSK decoding experiment. (a) Data received during 10 time frames. The color legend is shown on top. Note that only the real part is plotted. (b) Received data of the first time frame only, $t = 1$. (c) Received data of $t = 1$ plus virtual patterns (white dots) obtained by flipping the received data. (d) Clustering results obtained by OMGP. (e) Clustering results obtained by GP-BTT.

amplitude received at each frequency are due to the wireless channel's frequency selectivity. Variations in time also occur, most importantly due to movement of the transmitter or the receiver, or due to changes in the wireless channel, though these changes are much smoother.

In order to restore the original symbols from the received signal, the wireless channel has to be identified. This is achieved through the transmission of a known sequence of "pilot" BPSK symbols during the first two time frames, denoted the "long-training sequence". In this experiment we will show that GP-BTT is capable of blindly recovering the pilot symbol sequence. In other words, it allows to estimate the wireless

TABLE II
NUMBER OF SYMBOL ERRORS IN THE BPSK DECODING EXPERIMENT AT DIFFERENT SIGNAL-TO-NOISE RATES.

frames	SNR	OMGP	GP-BTT
t=1	10 dB	10	2
	15 dB	0	0
t=[1,2]	10 dB	n/a	1
	15 dB	n/a	0

channel, based only on an unknown sequence of received symbols. This sequence, which is transmitted during the pilot time frames, can be used to transmit additional information, effectively raising the information load of WLAN, and it is the scope of a parallel ongoing research project.

We captured WLAN data packets using a wireless communication test bed in a realistic indoor environment (see [29] for a more detailed description of the test bed). The signal-to-noise ratio (SNR) varied between 10 dB and 15 dB. Specifically, we captured 100 data packets at 10 db SNR and another 100 at 15 dB SNR. An example of one of the received pilot sequences at $t = 1$ is shown in Fig. 7(b). We process the data as follows. If the inverted BPSK sequence were transmitted, the received signal would be the negative of the true received signal, due to the linearity of the channel's operation [30]. Therefore, we add the negative signal as a set of "virtual" patterns to obtain a scenario that can be clustered by GP-BTT, see Fig. 7(c). Once these data are correctly clustered, it is possible to retrieve the labels of the transmitted data (the black dots in Fig. 7), which correspond to the transmitted byte sequence.

From each of the captured packets we took the first frame ($t = 1$) and added virtual patterns to it, as described above. We applied OMGP from [13] and the proposed GP-BTT algorithm to cluster the resulting data, for each packet separately. Each algorithm used automatic hyperparameter learning. The clustering result then allows to determine the labels $\in \{-1, +1\}$ corresponding to the received data up to a sign ambiguity, which can be resolved by sending one known pilot symbol. The results are shown in Figs. 7(d) and (e). This procedure could allow to add up to 200 bytes of additional information payload to the 1.2 kbytes contained in the WLAN data packet [27].

The total number of symbol decoding errors for each algorithm are listed in Table II. We then repeat the experiment but now use data from both pilot symbol time frames ($t = 1, 2$) to perform clustering. The input data space of the clustering problem is now two-dimensional with input variables (t, f). GP-BTT can be applied without any modification to such data. On the other hand, the implementation of OMGP in [13] is suitable only for sequential input data, and could therefore not be applied to this scenario.

E. Model estimation with multiple input dimensions

In the final experiment we visit a more general setting in which the latent functions do not necessarily represent a motion model. This corresponds to the most general case of data association, in which it is only assumed that the different data are described by different models. A motivation for this scenario is discussed in [6].

Specifically, we consider a problem in which the input space is two-dimensional. The latent functions chosen in this experiment are the two smooth surfaces depicted in Fig. 8(a). We assume that the measurement process yields observations that sample both functions jointly, and that the nature of the measurement process does not allow to determine in which order both observations were taken. In other words, at each location of the input space two samples are obtained, but it is not known to which source each of them corresponds. Note that, in the case of a one-dimensional input space this description corresponds to the standard multi-target tracking problem. Finally some noise is added to the observations. The observed values are graphically represented in Fig. 8(b). We used a grid of grid of 25 by 25 points to sample them.

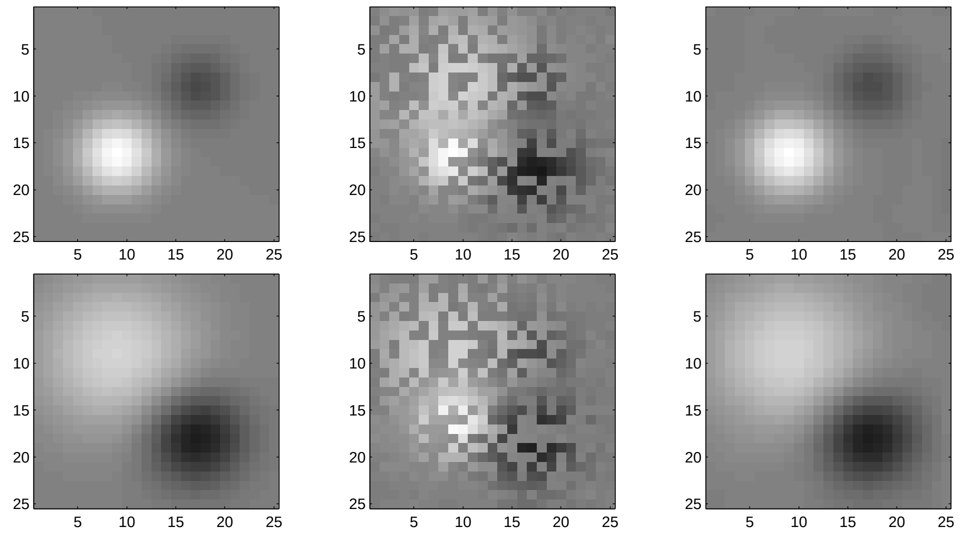
To the observed data we apply the GP-BTT algorithm with hyperparameter learning (see Section VI), for 25 different candidate pairs $\{\ell, \mathrm{SNR}\}$. The maximal NLML is obtained for the hyperparameters $\ell = 1.01$, $\mathrm{SNR} = 20.34$ dB. The retrieved latent surfaces are shown in Fig. 8(c). As can be observed, they correspond closely to the true hidden surfaces, i.e., GP-BTT has disambiguated the latent functions that model the observations. This demonstrates that GP-BTT is capable of solving the data association problem and de-noising the observed data.

VIII. CONCLUSIONS AND FUTURE DIRECTIONS

This work proposes a Bayesian model for data association that enforces the grouping of observations in smooth trajectories by using GP priors. A simple expression for the evidence of this model is provided, such that the suitability of any candidate set of labellings and hyperparameters can be evaluated in closed form. Using this idea, we can turn the data association problem into an optimization problem.

Evidence maximization for this model is a challenging task, especially when considering that the number of possible labellings grows exponentially with the number of observations. One of the main results of this paper is the reduction of this optimization problem, under some additional assumptions, to a constrained Max K-Section problem, which can be further reduced to a MaxCut problem when the number of sources is $K = 2$. This latter case allows to be efficiently and accurately solved using an SDP relaxation, yielding the proposed GP-BTT algorithm. We believe that the more general case in which $K > 2$ can be solved by using the constrained Max K-Section representation provided in this paper and SDP relaxations, though we leave this as an open problem.

The use of the Bayesian framework enables us to use the same objective function to perform model selection: Discrete optimization over labellings can be interleaved with continuous, nonlinear optimization over hyperparameters, or a grid search can be used. In both cases, the evidence is a sound criterion that allows direct comparison between any two candidate solutions. This is in contrast with other tracking methods in which parameters must be set by cross validation or trial-and-error.



(a) Source states. (b) Noisy, mixed observations. (c) Recovered states.

Fig. 8. Latent functions and observations of the data association experiment from Section VII-E. (a) Left column: the true underlying functions. (b) Middle column: observations, including measurement noise. (c) Right column: the function surfaces as retrieved by the proposed GP-BTT algorithm.

Although the equivalence between bi-target tracking and the MaxCut problem is an interesting theoretical result by itself, we have performed experiments to compare its practical performance against existing tracking algorithms. In our experiments, the competing methods had either more information than GP-BTT (for instance a more detailed, ground truth model, of the underlying dynamics of the sources) or were specifically tailored to the problem at hand (for instance discrete-valued observations coming from a sensor grid), yet produced similar or worse results at a higher computational cost. GP-BTT is fast, requires few hyperparameters, is very robust against model misspecification and it is capable of formulating an (approximate) offline solution that considers all possible data associations. Furthermore, as described in Section VII-E, it can be applied to problems with more than one *input* dimension, extending the concepts of data association and tracking beyond time-based concepts, which is how they have been conceived so far.

Naïve implementation of GPs limits their applicability to only a few thousand data samples. However, recent advances in sparse approximations (for instance [31], [32], [33]) should enable our approach to be applied to much larger data sets.

Acknowledgments

The authors would like to thank Jaspreet Singh, Samsung Telecommunications America, for making available the source code of the ClusterTrack algorithm. The authors would also like to thank Óscar González Fernández, University of Cantabria, Spain, for providing the data of the BPSK decoding experiment.

References

[1] Y. Bar-Shalom, *Tracking and data association*. Academic Press Professional, Inc. San Diego, CA, USA, 1987.

[2] I. Cox, "A review of statistical data association techniques for motion correspondence," *International Journal of Computer Vision*, vol. 10, no. 1, pp. 53–66, 1993.

[3] J. Singh, U. Madhow, S. Suri, and R. Cagley, "Multiple target tracking with binary proximity sensors," *ACM Transactions on sensor networks*, accepted for publication.

[4] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE Journal of Oceanic Engineering*, vol. 8, no. 3, pp. 173 – 184, jul 1983.

[5] S. Ullman, *The interpretation of visual motion*. M.I.T. Press, Cambridge, MA, USA, 1979.

[6] V. Cherkassky and Y. Ma, "Multiple model regression estimation," *IEEE Transactions on Neural Networks*, vol. 16, no. 4, pp. 785 –798, july 2005.

[7] D. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843 – 854, 1979.

[8] R. Karlsson and F. Gustafsson, "Monte Carlo data association for multiple target tracking," *IEEE International Seminar on Target Tracking: Algorithms and Applications*, vol. 1, p. 13, 2001.

[9] D. Chetverikov and J. Verestói, "Feature point tracking for incomplete trajectories," *Computing*, vol. 62, no. 4, pp. 321–338, 1999.

[10] C. Veenman, M. Reinders, and E. Backer, "Resolving motion correspondence for densely moving points," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 1, pp. 54–72, 2001.

[11] V. Nagarajan, M. Chidambara, and R. Sharma, "Combinatorial problems in multitarget tracking - a comprehensive solution," *IEE Proceedings-F: Communications, Radar and Signal Processing*, vol. 134, no. 1, pp. 113 –118, Feb. 1987.

[12] I. Cox and S. Hingorani, "An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 138 –150, Feb. 1996.

[13] M. Lázaro-Gredilla, S. Van Vaerenbergh, and N. D. Lawrence, "Overlapping mixtures of gaussian processes for the data association problem," *Pattern Recognition*, vol. 45, no. 4, pp. 1386 – 1395, 2012.

[14] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[15] C. E. Rasmussen, "Evaluation of Gaussian processes and other methods for non-linear regression," Ph.D. dissertation, University of Toronto, 1996.

[16] V. Tresp, "A Bayesian committee machine," *Neural Computation*, vol. 12, pp. 2719–2741, 2000.

[17] C. E. Rasmussen and Z. Ghahramani, "Infinite mixtures of Gaussian process experts," in *Advances in Neural Information Processing Systems 14*. MIT Press, 2002, pp. 881–888.

[18] E. Meeds and S. Osindero, "An alternative infinite mixture of Gaussian process experts," in *Advances in Neural Information Processing Systems 18*. MIT Press, 2006, pp. 883–890.

[19] C. Yuan and C. Neubauer, "Variational mixture of Gaussian process experts," in *Advances in Neural Information Processing Systems 21*, 2009, pp. 1897–1904.

[20] B. Schölkopf and A. J. Smola, *Learning with Kernels*. Cambridge, MA, USA: The MIT Press, 2002.
[21] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
[22] S. Mahajan and J. Ramesh, "Derandomizing semidefinite programming based approximation algorithms," in *Proc. 36th Ann. IEEE Symp. on Foundations of Comput. Sci.* IEEE Computer Society, 1995, pp. 162–169.
[23] A. Frieze and M. Jerrum, "Improved approximation algorithms for max k-cut and max bisection," *Algorithmica*, vol. 18, pp. 67–81, 1997.
[24] G. Andersson, "An approximation algorithm for max p-section," in *Proc. 16th Ann. Symp. on Theoretical Aspects of Comput. Sci*, ser. Lecture Notes in Comput. Sci. 1563. Springer-Verlag, 1999, pp. 237–247.
[25] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *J. ACM*, vol. 42, pp. 1115–1145, 1995.
[26] W. Kim, K. Mechitov, J.-Y. Choi, and S. Ham., "On target tracking with binary proximity sensors," in *Fourth International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 2005, pp. 301–308.
[27] "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: High-speed physical layer in the 5 GHz band," *IEEE Standard 802.11a-1999*, pp. 1–91, 1999.
[28] J. Proakis, *Digital Communications*. McGraw-Hill, 1995.
[29] J. Gutiérrez, Ó. González, J. Pérez, D. Ramírez, L. Vielva, J. Ibáñez, and I. Santamaría, "Frequency-domain methodology for measuring MIMO channels using a generic test bed," *IEEE Trans. on Instrumentation and Measurement*, vol. 60, no. 3, pp. 827–838, Mar. 2011.
[30] S. Van Vaerenbergh, I. Santamaria, P. Barbano, U. Ozertem, and D. Erdogmus, "Path-based spectral clustering for decoding fast time-varying MIMO channels," in *IEEE International Workshop on Machine Learning for Signal Processing*. IEEE, 2009, pp. 1–6.
[31] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *Advances in Neural Information Processing Systems 18*. MIT Press, 2006, pp. 1259–1266.
[32] M. K. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *Proceedings of the 12th International Workshop on AI Stats*, 2009, pp. 567–574.
[33] M. Lázaro-Gredilla and A. Figueiras-Vidal, "Inter-domain Gaussian processes for sparse inference using inducing features," in *Advances in Neural Information Processing Systems 22*. MIT Press, 2010, pp. 1087–1095.

Miguel Lázaro-Gredilla (M'11) received the telecommunication engineering degree (Honors) from the University of Cantabria, Santander, Spain, and the Ph.D. degree (Honors) from the Universidad Carlos III de Madrid, Leganés, Spain, in 2004 and 2010, respectively. He has been a visiting researcher at the University of Cambridge, United Kingdom; the University of Manchester, United Kingdom; and the University of Cantabria. Currently, he is a visiting lecturer at Universidad Carlos III de Madrid. His current research interests include Gaussian Processes, Bayesian models, and approximate inference.

Steven Van Vaerenbergh (S'06–M'11) received the M.Sc. degree in electrical engineering from Ghent University, Belgium, in 2003, and the Ph.D. degree from the University of Cantabria, Spain, in 2010. He was a visiting researcher at the Computational NeuroEngineering Laboratory, University of Florida, Gainesville, in 2008. Currently, he is a postdoctoral associate with the Department of Telecommunications Engineering, University of Cantabria, Spain. His current research interests include machine learning and its applications to adaptive filtering, target tracking, and system identification.