Gaussian Mixture Penalty for Trajectory Optimization Problems

Cédric Rommel, Frédéric Bonnans, Pierre Martinon, Baptiste Gregorutti

Gaussian Mixture Penalty for Trajectory Optimization Problems

Cédric Rommel*
CMAP, Ecole Polytechnique, Palaiseau, 91128, FRANCE
INRIA, Palaiseau, 91120, FRANCE
Safety Line, Paris, 75015, FRANCE

Frédéric Bonnans † and Pierre Martinon ‡
CMAP, Ecole Polytechnique, Palaiseau, 91128, FRANCE
INRIA, Palaiseau, 91120, FRANCE

Baptiste Gregorutti §
Safety Line, Paris, 75015, FRANCE

We consider the task of solving an aircraft trajectory optimization problem where the system dynamics have been estimated from recorded data. Additionally, we want to avoid optimized trajectories that go too far away from the domain occupied by the data, since the model validity is not guaranteed outside this region. This motivates the need for a proximity indicator between a given trajectory and a set of reference trajectories. In this presentation, we propose such an indicator based on a parametric estimator of the training set density. We then introduce it as a penalty term in the optimal control problem. Our approach is illustrated with an aircraft minimal consumption problem and recorded data from real flights. We observe in our numerical results the expected trade-off between the consumption and the penalty term.

Introduction

Aircraft trajectory optimization is a complex class of problems that has been extensively studied with respect to different aspects, such as fuel consumption [1], flight time [2] and noise reduction [3], as well as in collision avoidance [4]. More recently, a particular framework has been considered in which the aircraft dynamics have been estimated from previous flights data [5, 6]. This setting raises the question of whether the optimized trajectory does not deviate too much from the validity region of the dynamics model, which corresponds to the area occupied by the data used to build it. Moreover, for practical purposes and namely acceptance by pilots and Air Traffic Control, optimized trajectories that do not look too unfamiliar are preferable. These questions may both be addressed by quantifying the closeness between the optimization solution and the set of real flights used to identify the model. In [7], a method relying on estimations of

*PhD Student, cedric.rommel@safety-line.fr.
†Senior Researcher, Team COMMANDS, frederic.bonnans@inria.fr.
‡Researcher, Team COMMANDS, pierre.martinon@inria.fr
§Research Manager, baptiste.gregorutti@safety-line.fr

the probability density functions of some flight variables conditionally to the aircraft altitude is proposed for this matter. The suggested indicator, called the Mean Marginal Likelihood (MML), is estimated in practice by aggregating several kernel density estimators (KDE).

As pointed out in this paper, although a good discriminating power can be achieved with the MML in one dimension, using the KDE as marginal density estimator leads to poor performance when applied to more than one variable. Building the MML with a parametric density estimator could help dealing with such undesired behavior, since these are known to be less sensitive to dimensionality increase.

Furthermore, instead of just assessing simulated trajectories after the optimization, one could want to use such a criterion to penalize the optimal control problem, in order to obtain more acceptable trajectories directly. From a practical perspective, the nonparametric nature of the KDE-based MML from [7] makes it unsuitable for generic optimal control solvers such as Bocop [8], which require the objective and constraint functions to be compatible with automatic differentiation tools. It is in this context that we suggest an adaptation of the Mean Marginal Likelihood using Gaussian mixture models as marginal density estimator.

In what follows, we will first introduce the optimal control problem with identified dynamics and add to it a penalty term based on the parametric Mean Marginal Likelihood. After recalling the derivation of such a quantity, more details concerning the estimation of the Gaussian mixture models will be given. Finally, numerical results based on real aircraft data will be used to compare the performances of our MML extension to the original nonparametric version from [7]. They will also serve to illustrate the usefulness of our approach as a new trajectory optimization penalty.

Aircraft Trajectory Optimization as an Identified Optimal Control Problem

We consider an aircraft in climb phase, modeled as a dynamical system of state variables $\boldsymbol{x} \in \mathbb{X} \subset W^{1,\infty}(0, t_f; \mathbb{R}^{d_x})$ and control variables $\boldsymbol{u} \in \mathbb{U} \subset L^{\infty}(0, t_f; \mathbb{R}^{d_u})$. In such a context, the problem of optimizing the trajectory over a certain horizon $t_f > 0$ may be seen as a nonlinear constrained optimal control problem of the following form:

$$
\begin{aligned}
&\min_{(\boldsymbol{x},\boldsymbol{u})\in\mathbb{X}\times\mathbb{U}} \int_0^{t_f} C(t, \boldsymbol{u}(t), \boldsymbol{x}(t))dt, \\
&\text{s.t.} \begin{cases}
\dot{\boldsymbol{x}}(t) = \hat{g}(t, \boldsymbol{u}(t), \boldsymbol{x}(t)), \quad \text{for a.e. } t \in [0, t_f], \\
\boldsymbol{u}(t) \in U_{ad}, \quad \boldsymbol{x}(t) \in X_{ad}, \quad \text{for a.e. } t \in [0, t_f], \\
\Phi(\boldsymbol{x}(0), \boldsymbol{x}(t_f)) \in K_\Phi, \\
c_j(t, \boldsymbol{u}(t), \boldsymbol{x}(t)) \leq 0, \quad j = 1, \ldots, n_c, \quad \text{for all } t \in [0, t_f],
\end{cases}
\end{aligned} \tag{1}
$$

where $C \in C^2$ is the running cost, $\hat{g} \in C^1$ if the dynamics functions, $U_{ad} \subset \mathbb{R}^{d_u}$ and $X_{ad} \subset \mathbb{R}^{d_x}$ are the admissible (closed) sets for the controls and states, $\Phi \in C^1$ is the initial-final state constraint function, K_Φ is a nonempty closed

convex set of $\mathbb{R}^{n_\Phi}$ and $\{c_j\}_{j=1}^{n_c} \in (C^1)^{n_c}$ are the path constraint functions.

The function $\hat{g}$ from (1) reflects the fact that we want the optimized trajectory to respect some estimated model of the true aircraft dynamics, denoted by g. Assuming that one has access to some dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{u}_i, \dot{\boldsymbol{x}}_i)\}_{i=1}^m$ of previous flights, it may be used to build $\hat{g}$ (see e.g. [5, 6]). Note that, in a general setting, the dynamics model $\hat{g}$ can only be guaranteed to accurately approximate g close to the region occupied by the data. Hence, it seems desirable to prevent the solution of (1) from going too far away from $\mathcal{D}$. This is even more true for practical applications with high safety requirements such as airplane flights, since the optimized trajectories will have to be accepted by the Air Traffic Control and by the pilots supposed to fly them. Indeed, avoiding solutions outside some flight envelope of "realistic" trajectories should help on that.

This can be achieved by adding to (1) a flight domain as a hard constraint on the states and controls $(\boldsymbol{x}, \boldsymbol{u})$. Another more flexible possibility is to introduce a weight $\lambda > 0$ and a penalty term $\mathrm{Pen}_{\hat{g}}(\boldsymbol{u}, \boldsymbol{x})$ to the objective function:

$$\min_{(\boldsymbol{x},\boldsymbol{u})\in\mathbb{X}\times\mathbb{U}} \int_0^{t_f} C(t, \boldsymbol{u}(t), \boldsymbol{x}(t))dt + \lambda\, \mathrm{Pen}_{\hat{g}}(\boldsymbol{u}, \boldsymbol{x}),$$

$$\text{s.t.} \begin{cases} \dot{\boldsymbol{x}}(t) = \hat{g}(t, \boldsymbol{u}(t), \boldsymbol{x}(t)), \quad \text{for a.e. } t \in [0, t_f], \\ \boldsymbol{u}(t) \in U_{ad}, \quad \boldsymbol{x}(t) \in X_{ad}, \quad \text{for a.e. } t \in [0, t_f], \\ \Phi(\boldsymbol{x}(0), \boldsymbol{x}(t_f)) \in K_\Phi, \\ c_j(t, \boldsymbol{u}(t), \boldsymbol{x}(t)) \leq 0, \quad j = 1, \ldots, n_c, \quad \text{for all } t \in [0, t_f]. \end{cases} \tag{2}$$

In the following section we present a candidate penalty $\mathrm{Pen}_{\hat{g}}(\boldsymbol{u}, \boldsymbol{x})$.

Solution Assessment Using the Mean Marginal Likelihood Estimator

Numerical experiments presented in [7] show that the Mean Marginal Likelihood (MML) is a good discrepancy measure for quantifying the distance between an optimized climb trajectory and a set of real climb profiles. In this section, we present formally the main ideas of this method, keeping in mind that we want to use it as a penalty function $\mathrm{Pen}_{\hat{g}}(\boldsymbol{u}, \boldsymbol{x})$ for the control problem (2).

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, $\mathbb{T}$ be an interval of $\mathbb{R}_+$ and E be a compact set of $\mathbb{R}^d$, $d \in \mathbb{N}^*$. We denote by $z \in C(\mathbb{T}, E)$ some variable of interest, which can be formed by some of the components of the vector-valued functions $\boldsymbol{x}$ and $\boldsymbol{u}$. Let $Z = (Z_t)_{t\in\mathbb{T}}$ be a continuous random process from Ω to $C(\mathbb{T}, E)$ of distribution μ. For any $t \in \mathbb{T}$, we denote by μ_t the marginal distribution of Z at time t, whose probability density function $f_t \in L^1(E, \mathbb{R}_+)$ is called herefater the *marginal density* at time t. Let $\boldsymbol{y} \in C(\mathbb{T}, E)$ be an arbitrary trajectory of interest that we would like to compare to a set of m curves $\mathcal{T} = \{z^1, \ldots, z^m\} \subset C(\mathbb{T}, E)$ sampled from μ.

Up to a certain continuous scaling map $\psi : L^1(E, \mathbb{R}_+) \times \mathbb{R} \to [0; 1]$, the MML corresponds to the average over t of

the marginal density functions f_t evaluated on the points of $\boldsymbol{y}$:

$$\mathrm{MML}(Z, \boldsymbol{y}) = \frac{1}{t_f} \int_0^{t_f} \psi[f_t, \boldsymbol{y}(t)]dt. \tag{3}$$

Two possible choices for the scaling map were proposed in [7]:

- the normalized density

$$\psi[f_t, \boldsymbol{y}(t)] := \frac{\boldsymbol{y}(t)}{\max\limits_{z \in E} f_t(z)}, \tag{4}$$

- and the confidence level

$$\psi[f_t, \boldsymbol{y}(t)] := \mathbb{P}\left(f_t(Z_t) \leq f_t(\boldsymbol{y}(t))\right) \tag{5}$$

Assuming that all trajectories are sampled at variable discrete times

$$\{(t_j^r, z_j^r)\}_{\substack{1 \leq j \leq n \\ 1 \leq r \leq m}} \subset \mathbb{T} \times E, \qquad z_j^r := \boldsymbol{z}^r(t_j^r), \tag{6}$$

$$\{(\tilde{t}_j, y_j)\}_{j=1}^{\tilde{n}} \subset \mathbb{T} \times E, \qquad y_j := \boldsymbol{y}(\tilde{t}_j), \tag{7}$$

the MML can be estimated by using a Riemann sum in order to aggregate consistent estimators $\hat{f}_{\tilde{t}_j}^m$ of the marginal densities $f_{\tilde{t}_j}$:

$$\mathrm{EMML}_m(Z, \boldsymbol{y}) := \frac{1}{t_f} \sum_{j=1}^{\tilde{n}} \psi[\hat{f}_{\tilde{t}_j}^m, y_j]\Delta\tilde{t}_j. \tag{8}$$

The estimation of the marginal densities is discussed in the following section.

A Parametric Marginal Density Estimator

The Gaussian Mixture Model

Marginal density estimation can be done by uniformly partitioning the space of times $\mathbb{T}$ into *bins* and building standard density estimators using the data points whose sampling times fall in each bin. A special instance of kernel density estimators (KDE) were proposed for this task in [7], which are nonparametric. Despite the great flexibility of the latter, parametric models may be better suited in some contexts, such as solving numerically penalized problems of the form of (2). For this reason, we considered in this paper the use of Gaussian mixture models (GMM) for the marginal density estimation.

Indeed, despite being straightforward parametric models, Gaussian mixtures are still quite flexible, being interpretable as simpler versions of Gaussian kernel density estimators*. These models are known for being well-suited for problems in which the data falls into a small number of cluster. In our case, we indeed observe that, for a given zone in time

*While KDE's place a Gaussian density around each observation, GMM places fewer Gaussians around estimated centers.

during climb phase, different trajectories from the same aircraft will often concentrate around a small number "standard" values.

In this model, for some time $t \in \mathbb{T}$, the marginal density is assumed to be a finite convex combination of K terms

$$\forall z \in E, \qquad f_h(z) = \sum_{k=1}^{K} w_{t,k}\,\phi(z, \mu_{t,k}, \Sigma_{t,k}), \qquad \text{with } \sum_{k=1}^{K} w_{t,k} = 1, \tag{9}$$

where all weights $w_{t,k}$ are nonnegative and $\phi(\cdot, \mu, \Sigma)$ denotes a d-dimensional Gaussian density function with mean $\mu \in \mathbb{R}^d$ and covariance matrix $\Sigma \in \mathcal{S}_d(\mathbb{R})$:

$$\phi(z, \mu, \Sigma) := \frac{1}{\sqrt{(2\pi)^d \det \Sigma}} \exp\left(-\frac{1}{2}(z-\mu)^\top \Sigma^{-1}(z-\mu)\right). \tag{10}$$

Let $\{z_1, \ldots, z_N\} \subset E$ denote the subsample of observations falling in the bin containing t, considered as a neighborhood. As the means and covariances of the K components are unknown, they need to be inferred using the data, which is usually done through maximum likelihood estimation. Had we one component, i.e. $K = 1$, the log-likelihood of a sample of observations $\{z_i\}_{i=1}^N$ would be

$$\sum_{i=1}^{N} -\frac{1}{2}\left(\log(2\pi) - \log\det\Sigma_{h,1} - (z_i - \mu_{h,1})^\top \Sigma_{h,1}^{-1}(z_i - \mu_{h,1})\right), \tag{11}$$

which is maximized by the sample mean and covariance:

$$\hat{\mu}_{h,1} = \frac{1}{N}\sum_{i=1}^{N} z_i, \qquad \hat{\Sigma}_{h,1} = \frac{1}{N}\sum_{i=1}^{N}(z_i - \hat{\mu}_{h,1})(z_i - \hat{\mu}_{h,1})^\top. \tag{12}$$

In the general case $K > 1$ there is no closed-form for the maximum likelihood estimates, which is why an Expectation-Maximization (EM) algorithm is usually preferred.

The Expectation-Maximization Algorithm

The EM is a well-established optimization algorithm for maximum likelihood estimation originally proposed by [9]. The key of the EM procedure is to suppose the existence of a hidden random variable J valued on $\{1, \ldots, K\}$, such that the i.i.d sample of its observations $\{J_i\}_{i=1}^N$, "labels" the observations of Z_t: if $J_i = k$, then the i^{th} observation z_i was drawn from the k^{th} Gaussian component. In this case, we can interpret $\phi(z_i, \mu_{t,k}, \Sigma_{t,k})$ as the probability density of $Z_{h,i}$ conditioned by $J_i = 1$ and the weight $w_{t,k}$ as the prior probability of being drawn from the k^{th} component $\mathbb{P}(J_i = k)$. Furthermore, the usefulness of the unobserved variables J_i is intuitive since, if we knew for each sample z_i which component generated it, we could just group them by component and compute the sample means and covariances separately as done in (12).

That being said, the EM falls into two main steps. Starting from an initial guess of parameters $\hat{\theta}_t = (\hat{w}_{t,k}, \hat{\mu}_{t,k}, \hat{\Sigma}_{t,k})_{k=1}^K$, the first step corresponds to the computation of the posterior probabilities of the labels $\mathbb{P}(J_i = k|\hat{\theta}_t, Z_h)$, usually called *responsibilities*:

$$\hat{\pi}_{k,i} := \mathbb{P}(J_i = k|\hat{\theta}_t, Z_h) = \frac{\hat{\mu}_{t,k}\phi(z_i, \hat{\mu}_{t,k}, \hat{\Sigma}_{t,k})}{\sum_{j=1}^N \hat{w}_{t,k}\phi(z_j, \hat{\mu}_{t,k}, \hat{\Sigma}_{t,k})}, \qquad k = 1, \dots, K, \quad i = 1, \dots, N. \tag{13}$$

This step is called the *Expectation* step since it estimates a discrete probability distribution which is later used to compute the expectation of the log-likelihood to be maximized. As explained in the appendix, this criteria is a lower bound of the log-likelihood, which is the main idea of the procedure. Its maximization is carried in the second step, called the *Maximization* step, which brings the weights to be approximated by the empirical mean of the responsibilities

$$\hat{w}_{t,k} = \frac{1}{N}\sum_{i=1}^N \hat{\pi}_{k,i}, \tag{14}$$

and the means and covariances to be the weighted maximum likelihood estimates:

$$\hat{\mu}_{t,k} = \frac{\sum_{i=1}^N \hat{\pi}_{k,i} z_i}{\sum_{i=1}^N \hat{\pi}_{k,i}}, \qquad \hat{\Sigma}_{t,k} = \frac{\sum_{i=1}^N \hat{\pi}_{k,i}(z_i - \hat{\mu}_{t,k})(z_i - \hat{\mu}_{t,k})^\top}{\sum_{i=1}^N \hat{\pi}_{k,i}}. \tag{15}$$

These steps are then repeated until convergence. For this reason, the EM can be seen as an alternating optimization algorithm in which we successively adjust the weights probabilities and the parameters values. It has been shown in [9] that such a procedure converges linearly to a local maximum of the likelihood and [10] proved that it is competitive to other superlinear and quadratic optimization algorithms.

Application to an Aircraft Minimal-Consumption Problem

Experiments Description

In this section, we test the proposed approach with real aircraft data. First, we compare the suggested GMM-based Mean Marginal Likelihood criterion to the original nonparametric kernel-based MML from [7]. Then, we solve a series of trajectory optimization problems penalized with such an indicator and assess the obtained results.

Data Description

In all experiments presented, the MML estimators were computed based on a training set of 424 different flights from the same medium haul aircraft, corresponding to a total of 334 531 observation points. This dataset was extracted from the *Quick Access Recorder* (QAR) and is exactly the same as in [7]. We only used signals sampled at 1 Hz of 5 variables, namely the altitude h, the true airspeed V (or TAS), the path angle γ, the angle of attack α and the average engines turbofan speed N_1. Some of these signals were not directly available and were computed from other measurements using

standard flight mechanics formulas (see e.g. [5]). Furthermore, we kept only the portion of these signals corresponding to the climb phase, starting from 5000 ft. This same dataset was used to estimate the aircraft dynamics with the methods described in [5]. All optimization problems solved in this section used this same dynamics model.

Concerning the comparisons made between the parametric and nonparametric versions, the test set used to compute the MML scores was also the same as in [7]:

1) 50 real flights, extracted from the training set prior to training;
2) 50 simulated trajectories obtained by using Bocop [8] to solve problem (1), with operational constraints keeping the resulting speed V and N_1 between reasonable bounds;
3) and another 50 simulated trajectories optimized without the operational constraints.

Choice of the "time" variable

As all the trajectories considered are climb profiles, a change of variable is used to parametrize all functions using the altitude h, which plays the role of "time" here. Indeed, this is quite a standard practice in trajectory optimization since h is nondecreasing during this phase of flight and all other variables strongly depend on it.

Algorithm Settings

For all MML scores computation, the partitioning of the altitude was carried exactly as in [7], i.e.:

- between 1524 and 3000m and between 4000 and 12000m, the altitudes were divided homogeneously into intervals of size $b_m^{(1)} = 21\text{m} \simeq 1/\sqrt{m}$;
- between 3000 and 4000m we used an interval size $b_m^{(2)} = 10\text{m} \simeq b_m^{(1)}/2$.

The altitudes between 3000 and 4000m define a zone where climb-steps are common. Hence, faster variations w.r.t. the altitude can be observed in this region (see e.g. figures 2), which explains why a finer partition was preferred.

All self-consistent kernel density estimations were carried using the same implementation as in [7], i.e. the python library fastKDE [11, 12]. The parameters of the Gaussian mixtures were learned using the scikit-learn library [13], which implements the EM algorithm described in section IV.B. The number of components was chosen between 1 and 5 using the Bayesian information criteria (BIC) [14]. Default settings were kept for the initialization: initial weights, means and covariances were set by solving a K-means clustering problem 10 times using Lloyd's algorithm [15] initialized with centroids drawn randomly from the dataset.

Concerning the optimal control problems (1) and (2), both were set with the estimated dynamics and solved using Bocop [8]. The altitude was used as independent variable (i.e 'time') with 1000 steps for all optimizations. The initial and final conditions of the penalized control problems were set to match 20 flights randomly drawn from the training dataset.

Numerical Comparison Between MML using Gaussian Mixture and Self-Consistent Kernel Estimators

Tables 1a and 1b contain the MML scores using normalized density (4) and confidence level (5) as scaling maps for both parametric and nonparametric formulations. These scores were computed in a 1-dimensional setting: each variable of interest (V, γ, α and N_1) was processed independently. We see that the values returned by both implementations of the MML estimator are statistically the same for all four variables considered, with both scaling maps. As a consequence, the discriminating power of the GMM-based method is as good as the KDE-based one.

Table 1 Average and standard deviation of the Mean Marginal Likelihood scores in 1-dimensional setting using confidence level and normalized density for 50 real flights (*Real*), 50 optimized flights with operational constraints (*Opt1*) and 50 optimized flights without constraints (*Opt2*).

(a) Self-consistent kernel estimator

Var.	Confidence level			Normalized Density		
	Real	Opt1	Opt2	Real	Opt1	Opt2
V	0.52 ± 0.16	0.38 ± 0.14	0.15 ± 0.09	0.63 ± 0.16	0.45 ± 0.16	0.17 ± 0.10
γ	0.54 ± 0.09	0.24 ± 0.12	0.22 ± 0.09	0.67 ± 0.09	0.33 ± 0.17	0.29 ± 0.11
α	0.53 ± 0.06	0.08 ± 0.05	0.02 ± 0.01	0.65 ± 0.07	0.10 ± 0.06	0.02 ± 0.01
N_1	0.47 ± 0.24	0.71 ± 0.00	0.03 ± 0.01	0.57 ± 0.27	0.83 ± 0.01	0.04 ± 0.02
Mean	0.52 ± 0.07	0.35 ± 0.06	0.10 ± 0.02	0.63 ± 0.07	0.43 ± 0.08	0.13 ± 0.02

(b) GMM

Var.	Confidence level			Normalized Density		
	Real	Opt1	Opt2	Real	Opt1	Opt2
V	0.51 ± 0.15	0.37 ± 0.13	0.15 ± 0.08	0.61 ± 0.16	0.44 ± 0.15	0.17 ± 0.09
γ	0.53 ± 0.09	0.24 ± 0.13	0.22 ± 0.08	0.67 ± 0.08	0.33 ± 0.17	0.29 ± 0.11
α	0.54 ± 0.06	0.08 ± 0.05	0.01 ± 0.01	0.67 ± 0.06	0.10 ± 0.07	0.02 ± 0.01
N_1	0.47 ± 0.23	0.71 ± 0.01	0.03 ± 0.01	0.55 ± 0.26	0.83 ± 0.01	0.02 ± 0.02
Mean	0.51 ± 0.06	0.35 ± 0.06	0.10 ± 0.02	0.63 ± 0.07	0.42 ± 0.08	0.12 ± 0.02

Tables 2a and 2b contain the scores in a 2-dimensional setting: variables were grouped in two pairs, (V, γ) and (α, N_1), so that densities of variables of the same group are estimated jointly. The first group contains state variables, supposed to vary slowly, while the second contains control variables. As noted in [7], the scores obtained using the kernel estimator version are abnormally low, even for the real flights, and the three categories of test flights are not well separated. We suspect that this poor performance was caused by the high model-complexity of this nonparametric method, which makes it more sensitive to the curse of dimensionality. Following this line of thought, parametric models should have a better behavior in this context. The scores obtained by the GMM version seem to confirm this assumption. Indeed, the three groups of flights are well discriminated by the MML scores, being higher for the real flight and lower for the flights optimized without reasonable constraints. Moreover, the training of the GMM version took in average

only 1.9 seconds[†], which is more than 40 times faster than the kernel version (81.4 seconds in average). However, the variances observed in this 2-dimensional setting are larger than those obtained in dimension one. No discriminating advantage seems to exist in this context relative to the previous one, which is why variables scores were computed separately in the following section.

Table 2 Average and standard deviation of the Mean Marginal Likelihood scores in 2-dimensional setting using confidence level and normalized density for 50 real flights (*Real*), 50 optimized flights with operational constraints (*Opt1*) and 50 optimized flights without constraints (*Opt2*).

(a) Self-consistent kernel estimator

Var.	Confidence level			Normalized Density		
	Real	Opt1	Opt2	Real	Opt1	Opt2
(V, γ)	0.09 ± 0.05	0.11 ± 0.05	0.03 ± 0.03	0.05 ± 0.03	0.06 ± 0.03	0.01 ± 0.01
(α, N_1)	0.03 ± 0.02	0.01 ± 3e-3	0.01 ± 2e-3	0.02 ± 0.01	4e-3 ± 2e-3	3e-3 ± 1e-3
Mean	0.06 ± 0.03	0.06 ± 0.02	0.02 ± 0.01	0.03 ± 0.02	0.03 ± 0.02	0.01 ± 0.01

(b) GMM

Var.	Confidence level			Normalized Density		
	Real	Opt1	Opt2	Real	Opt1	Opt2
(V, γ)	0.52 ± 0.17	0.22 ± 0.10	0.07 ± 0.03	0.59 ± 0.14	0.30 ± 0.09	0.08 ± 0.08
(α, N_1)	0.93 ± 0.03	0.84 ± 0.01	0.82 ± 0.01	0.63 ± 0.12	0.12 ± 0.06	0.01 ± 0.01
Mean	0.73 ± 0.10	0.53 ± 0.05	0.45 ± 0.01	0.61 ± 0.12	0.21 ± 0.06	0.05 ± 0.04

Penalized Optimization Results

In order to check that the GMM-based Mean Marginal Likelihood can be used as a penalty term $\text{Pen}_{\hat{g}}(\boldsymbol{u}, \boldsymbol{x})$ in the optimal control problem (2), we tested different penalty weights ranging from 0 to 1000 for 20 initial and final conditions. Only the 1-dimensional MML was computed using the normalized density as scaling map. This choice was motivated by the results from the previous section showing that 1D computations allow a better discrimination and that both scaling maps lead to similar results, while the normalized density is easier to compute. Moreover, only the state variables V and γ were penalized in the problem, since penalizing the control variables led to convergence difficulties. The total fuel consumption and Mean Marginal Likelihood scores of the optimized flights, averaged over all 20 initializations, are plotted on figure 1. First, we confirm that the more we penalize, the more we consume and the higher the Mean Marginal Likelihood of the trajectory. Furthermore, we see that an MML score comparable to a real flight (i.e. > 0.6) is attainable with enough penalization, with and without operational constraints. Of course , this is achieved at the cost of a higher fuel consumption, which illustrates that a trade-off must be found for practical applications. We can also observe that with the highest values of λ, similar consumptions and MML scores are reached

[†]on a single 2.30 GHz core with 7.7 GB memory

for both types of constraints. Indeed, if we penalize too much we end up always finding a solution passing in the same narrow zone of highest densities, as depicted for one example on figure 2.

Conclusion

In this article we introduced a parametric version of the Mean Marginal Likelihood estimator from [7]. This new estimator was shown to lead to equivalent scores and discriminative power when used for quantifying the distance between a simulated climb trajectory and a set of real flights. It also overcomes the limitations of the former kernel-based estimator relative to multi-dimensional data. Furthermore, our numerical simulations show that our MML estimator is suitable for penalizing an optimal control problem, being a tool to obtain trajectories with good objective while still being acceptable for practical applications.

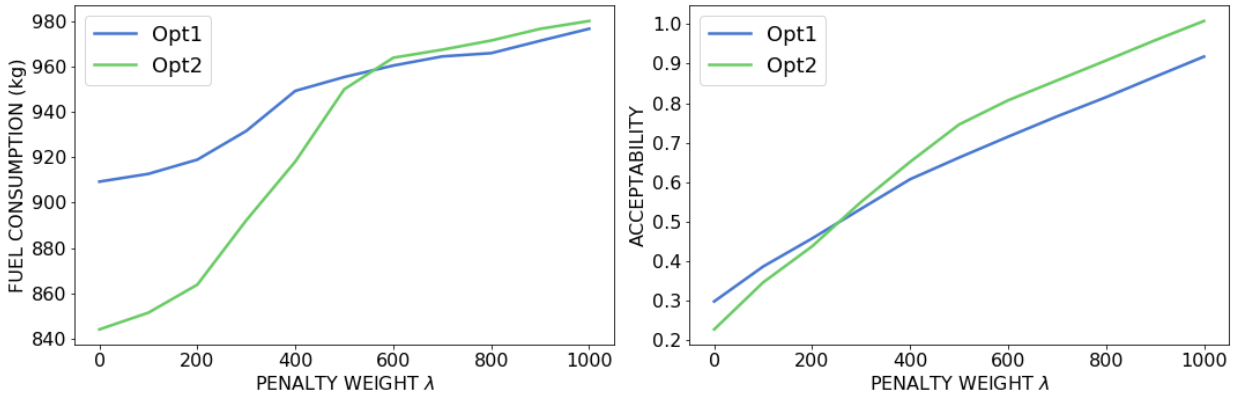
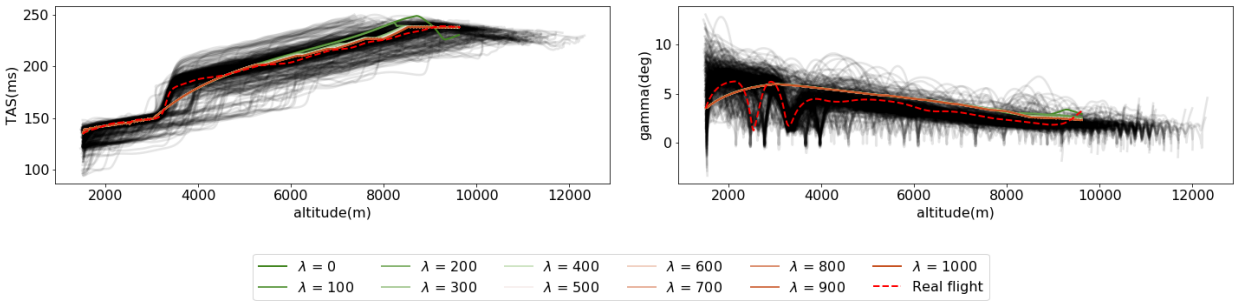
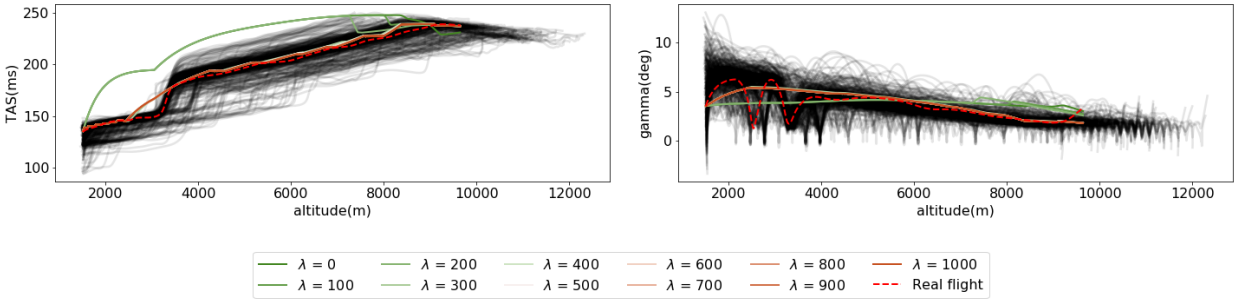


Fig. 1 Average over 20 flights of the total fuel consumption and MML score (called *acceptability* here) of optimized trajectories with varying MML-penalty weight λ.



(a) With operational constraints (Opt1).



(b) Without operational constraints (Opt2).

Fig. 2 Example of optimized flight with different MML-penalty weights λ. The dashed red lines correspond to the real trajectory used to define the initial and final conditions for the optimization.

References

[1] Cots, O., Gergaud, J., and Goubinat, D., "Direct and indirect methods in optimal control with state constraints and the climbing trajectory of an aircraft," *Optimal Control Applications and Methods*, Vol. 39, No. 1, 2018, pp. 281–301.

[2] Nguyen, N., "Singular arc time-optimal climb trajectory of aircraft in a two-dimensional wind field," *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2006, p. 6598.

[3] Khardi, S., Abdallah, L., Konovalova, O., and Houacine, M., "Optimal approach minimizing aircraft noise and fuel consumption," *Acta acustica united with Acustica*, Vol. 96, No. 1, 2010, pp. 68–75.

[4] Cafieri, S., Cellier, L., Messine, F., and Omheni, R., "Combination of optimal control approaches for aircraft conflict avoidance via velocity regulation," *Optimal Control Applications and Methods*, Vol. 39, No. 1, 2018, pp. 181–203.

[5] Rommel, C., Bonnans, J. F., Gregorutti, B., and Martinon, P., "Aircraft dynamics identification for optimal control," *Proceedings of the 7th European Conference for Aeronautics and Aerospace Sciences*, 2017.

[6] Rommel, C., Bonnans, J. F., Gregorutti, B., and Martinon, P., "Block sparse linear models for learning structured dynamical systems in aeronautics," , 2018.

[7] Rommel, C., Bonnans, J. F., Gregorutti, B., and Martinon, P., "Quantifying the Closeness to a Set of Random Curves via the Mean Marginal Likelihood," , 2018.

[8] Bonnans, J. F., Giorgi, D., Grelard, V., Heymann, B., Maindrault, S., Martinon, P., Tissot, O., and Liu, J., "Bocop – A collection of examples," Tech. rep., INRIA, 2017. URL `http://www.bocop.org`.

[9] Dempster, A. P., Laird, N. M., and Rubin, D. B., "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (methodological)*, 1977, pp. 1–38.

[10] Xu, L., and Jordan, M. I., "On convergence properties of the EM algorithm for Gaussian mixtures," *Neural computation*, Vol. 8, No. 1, 1996, pp. 129–151.

[11] O'Brien, T. A., Collins, W. D., Rauscher, S. A., and Ringler, T. D., "Reducing the Computational Cost of the ECF Using a nuFFT: A Fast and Objective Probability Density Estimation Method," *Computational Statistics & Data Analysis*, Vol. 79, 2014, pp. 222–234.

[12] O'Brien, T. A., Kashinath, K., Cavanaugh, N. R., Collins, W. D., and O'Brien, J. P., "A fast and objective multidimensional kernel density estimation method: fastKDE," *Computational Statistics & Data Analysis*, Vol. 101, 2016, pp. 148–160.

[13] Pedregosa, F., et al., "Scikit-learn: Machine Learning in Python," *JMLR*, Vol. 12, 2011, pp. 2825–2830.

[14] Schwarz, G., "Estimating the dimension of a model," *The Annals of Statistics*, Vol. 6, No. 2, 1978, pp. 461–464.

[15] Lloyd, S., "Least squares quantization in PCM," *IEEE transactions on information theory*, Vol. 28, No. 2, 1982, pp. 129–137.

[16] Sridharan, R., "An introduction to the EM algorithm, motivated by Gaussian mixture models," , 2014. URL http://people.csail.mit.edu/rameshvs/content/gmm-em.pdf.

Appendix

More details on the derivation of the EM algorithm

In this subsection we give more insight on the statistical origin of the EM algorithm. The explanations hereafter are strongly inspired by the excellent notes [16] towards which the reader is referred for further details.

We see from the expression of the Gaussian mixture density (9) that the log-likelihood of a sample of N observations $\{z_i\}_{i=1}^N$ writes:

$$\log f_h(z_1, \dots, z_N|\theta) = \sum_{i=1}^{N} \log\left(\sum_{k=1}^{K} w_{t,k}\phi(z_i, \mu_{t,k}, \Sigma_{t,k})\right). \tag{16}$$

It is the presence of the sum inside the logarithm that is problematic here. However, when we add the hidden variable J, we have a simpler expression for the log-likelihood relative to the distribution of the joint variable (Z_h, J):

$$\log p_{Z_h,J}\left(\{(z_i, J_i)\}_{i=1}^N|\theta\right) = \sum_{i=1}^{N}\sum_{k=1}^{K} \delta_k(J_i)\left(\log \phi(z_i, \mu_{t,k}, \Sigma_{t,k})\right), \tag{17}$$

where $\delta_k(x) = 1$ if $x = k$ and 0 otherwise, for $k = 1, \dots, K$. This corresponds to the remark that if we knew for each sample z_i which component generated it, we could just group them by component and compute the sample means and covariances separately as done in (12).

By writing f_h as the marginal over J of $p_{Z_h,J}$ and artificially introducing the density of p_J of J, we can write the log-likelihood (16) in terms of the expectation relative to J:

$$\begin{aligned}
\log f_h(z_1, \dots, z_N|\theta) &= \log \sum_{k=1}^{K} p_{Z_h,J}\left(\{(z_i, J_i = k)\}_{i=1}^N|\theta\right), \\
&= \log \sum_{k=1}^{K} \frac{p_{Z_h,J}\left(\{(z_i, J_i = k)\}_{i=1}^N|\theta\right)}{p_J(J_i = k)} p_J(J_i = k), \\
&= \log \mathbb{E}_J\left[\frac{p_{Z_h,J}\left(\{(z_i, J_i)\}_{i=1}^N|\theta\right)}{p_J(J_i)}\right], \\
&\geq \mathbb{E}_J\left[\log \frac{p_{Z_h,J}\left(\{(z_i, J_i)\}_{i=1}^N|\theta\right)}{p_J(J_i)}\right].
\end{aligned} \tag{18}$$

The lower-bound in (18) comes from Jensen's inequality. Using the definition of the conditional density, we can rewrite it as follows:

$$\begin{aligned}
\mathbb{E}_J\left[\log \frac{p_{Z_h,J}\left(\{(z_i, J_i)\}_{i=1}^N|\theta\right)}{p_J(J_i)}\right] &= \mathbb{E}_J\left[\log \frac{p_{J|Z_h}\left(J|\{z_i\}_{i=1}^N;\theta\right) f_h\left(\{z_i\}_{i=1}^N;\theta\right)}{p_J(J_i)}\right] \\
&= \log f_h\left(\{z_i\}_{i=1}^N;\theta\right) - \mathrm{KL}\left(p_J \parallel p_{J|Z_h}\left(\cdot|\{z_i\}_{i=1}^N;\theta\right)\right),
\end{aligned} \tag{19}$$

where KL denotes the Kullbeck-Liebler divergence between the posterior distribution of J and its true distribution. This

allows to interpret the Expectation step of the EM algorithm as the maximization of criteria (19) w.r.t. p_J, which is reached for $p_J = p_{J|Z_h}\left(\cdot|\{z_i\}_{i=1}^N;\theta\right)$.

Concerning the Maximization step, (18) can be written in the following way:

$$\mathbb{E}_J\left[\log\frac{p_{Z_h,J}\left(\{(z_i,J_i)\}_{i=1}^N|\theta\right)}{p_J(J_i)}\right] = \mathbb{E}_J\left[\log p_{Z_h,J}\left(\{(z_i,J_i)\}_{i=1}^N|\theta\right)\right] - \mathbb{E}_J\left[\log p_J(J_i)\right]. \tag{20}$$

We see that only the first term in (20) depends on θ and that it corresponds to the expectation of (17), which we know how to compute for give p_J.