Unsupervised aircraft trajectories clustering: a minimum entropy approach

Florence Nicol, Stéphane Puechmorel

Unsupervised aircraft trajectories clustering: a minimum entropy approach

Florence Nicol

Ecole Nationale de l'Aviation Civile
7, Avenue Edouard Belin,
F-31055 Toulouse FRANCE
Email: nicol@recherche.enac.fr

Stephane Puechmorel

Ecole Nationale de l'Aviation Civile
7, Avenue Edouard Belin,
F-31055 Toulouse FRANCE
Email: stephane.puechmorel@enac.fr

***Abstract*—Clustering is a common operation in statistics. When data considered are functional in nature, like curves, dedicated algorithms exist, mostly based on truncated expansions on Hilbert basis. When additional constraints are put on the curves, like in applications related to air traffic where operational considerations are to be taken into account, usual procedures are no longer applicable. A new approach based on entropy minimization and Lie group modeling is presented here, yielding an efficient unsupervised algorithm suitable for automated traffic analysis. It outputs cluster centroids with low curvature, making it a valuable tool in airspace design applications or route planning.**

Keywords–curve clustering; probability distribution estimation; functional statistics; minimum entropy; air traffic management.

I. Introduction

Clustering aircraft trajectories is an important problem in Air Traffic Management (ATM). It is a central question in the design of procedures at take-off and landing, the so called sid-star (Standard Instrument Departure and Standard Terminal Arrival Routes). In such a case, one wants to minimize the noise and pollutants exposure of nearby residents while ensuring runway efficiency in terms of the number of aircraft managed per time unit.

The same question arises with cruising aircraft, this time the mean flight path in each cluster being used to optimally design the airspace elements (sectors and airways). This information is also crucial in the context of future air traffic management systems where reference trajectories will be negotiated in advance so as to reduce congestion. A special instance of this problem is the automatic generation of safe and efficient trajectories, but in such a way that the resulting flight paths are still manageable by human operators. Clustering is a key component for such tools: major traffic flows must be organized in such a way that the overall pattern is not too far from the current organization, with aircraft flying along airways. The classification algorithm has thus not only to cluster similar trajectories but at the same time make them as close as possible to operational trajectories. In particular, straightness of the flight segments must be enforced, along with a global structure close to a graph with nodes corresponding to merging/splitting points and edges the airways.

II. Previous related work

Several well established algorithms may be used for performing clustering on a set of trajectories, although only a few of them were eventually applied in the context of air traffic. The spectral approach relies on trajectories modeling as vectors of samples in a high dimensional space, and uses random projections as a mean of reducing the dimensionality. The huge computational cost of the required singular values decomposition is thus alleviated, allowing use on real recorded traffic over several months. It was applied in a study conducted by the Mitre corporation on behalf of the Federal Aviation Authority (FAA) [**?**]. The most important limitation of this approach is that the shape of the trajectories is not taken into account when applying the clustering procedure unless a resampling procedure based on arclength is applied: changing the time parametrization of the flight paths will induce a change in the classification. Furthermore, there is no mean to put a constraint on the mean trajectory produced in each cluster: curvature may be quite arbitrary even if samples individually comply with flight dynamics.

Another approach is taken in [**?**], with an explicit use of an underlying graph structure. It is well adapted to road traffic as vehicles are bound to follow predetermined segments. A spatial segment density is computed then used to gather trajectories sharing common parts. For air traffic applications, it may be of interest for investigating present situations, using the airways and beacons as a structure graph, but will misclassify aircraft following direct routes which is quite a common situation, and is unable to work on an unknown airspace organization. This point is very important in applications since trajectory datamining tools are mainly used in airspace redesign. A similar approach is taken in [**?**] with a different measure of similarity. It has to be noted that many graph-based algorithms are derived from the original work presented in [**?**], and exhibit the aforementioned drawbacks for air traffic analysis applications.

An interesting vector field based algorithm is presented in [**?**]. A salient feature is the ability to distinguish between close trajectories with opposite orientations. Nevertheless, putting constraints on the geometry of the mean path in a cluster is quite awkward, making the method unsuitable for our application.

Due to the functional nature of trajectories, that are basically mappings defined on a time interval, it seems more appropriate to resort to techniques based on times series, as surveyed in [**?**], [**?**] or functional data statistics, with standard references [**?**], [**?**]. In both approaches, a distance between pairs of trajectories or, in a weaker form, a measure of similarity must be available. The algorithms of the first category are based on sequences, possibly in conjunction with dynamic time warping [**?**] while in the second samples are assumed to come

from an unknown underlying function belonging to a given Hilbert space. However, it has to be noticed that apart from this last assumption, both approaches yield similar end algorithms, since functional data revert for implementation to usual finite dimensional vectors of expansion coefficients on a suitable truncated basis. For the same reason, model-based clustering may be used in the context of functional data even if no notion of probability density exists in the original infinite dimensional Hilbert space as mentioned in[**?**]. A nice example of a model-based approach working on functional data is funHDDC [**?**].

III. Dealing with curve systems: a paradigm change

When working with aircraft trajectories, some specific characteristics must be taken into account. First of all, flight paths consist mainly of straight segments connected by arcs of circles, with transitions that may be assumed smooth up to at least the second derivative. This last property comes from the fact that pilot's actions result in changes on aerodynamic forces and torques and a straightforward application of the equations of motion. When dealing with sampled trajectories, this induces a huge level of redundancy within the data, the relevant information being concentrated around the transitions. Second, flight paths must be modeled as functions from a time interval $[a, b]$ to $\mathbb{R}^3$ which is not the usual setting for functional data statistics: most of the work is dedicated to real valued mappings and not vector ones. A simple approach will be to assume independence between coordinates, so that the problem falls within the standard case. However, even with this simplifying hypothesis, vertical dimension must be treated in a special way as both the separation norms and the aircraft maneuverability are different from those in the horizontal plane.

Finally, being able to cope with the initial requirement of compliance with the current airspace structure in airways is not addressed by general algorithms. In the present work, a new kind of functional unsupervised classifier is introduced, that has in common with graph-based algorithms an estimation of traffic density but works in a continuous setting. For operational applications, a major benefit is the automatic building of a route-like structure that may be used to infer new airspace designs. Furthermore, smoothness of the mean cluster trajectory, especially low curvature, is guaranteed by design. Such a feature is unique among existing clustering procedures. Finally, our Lie group approach makes easy the separation between neighboring flows oriented in opposite directions. Once again, it is mandatory in air traffic analysis where such a situation is common.

In the first section the notion of entropy of a curve system is introduced. The modeling of trajectories with a Lie group approach is then presented. The next two sections will show how to estimate Lie group densities and to cluster curves in this new setting. Finally, results on a synthetic example are briefly given and a conclusion is drawn.

IV. The entropy of a system of curves

Considering trajectories as mappings $\gamma\colon [t_0, t_1] \to \mathbb{R}^3$ induces a notion of spatial density as presented in [**?**]. Assuming that after a suitable registration process all flight paths $\gamma_i, i = 1, \ldots, N$ are defined on the same time interval $[0, 1]$ to Ω a domain of $\mathbb{R}^3$, one can compute an entropy associated with the system of curves using the approach presented in [**?**]. Let a system of curves $\gamma_1, \ldots, \gamma_N$ be given, its entropy is defined to be:

$$E(\gamma_1, \ldots, \gamma_N) = -\int_\Omega \tilde{d}(x) \log\left(\tilde{d}(x)\right) dx,$$

where the spatial density d is computed according to:

$$\tilde{d}\colon x \mapsto \frac{\sum_{i=1}^N \int_0^1 K\left(\|x - \gamma_i(t)\|\right) \|\gamma_i'(t)\| dt}{\sum_{i=1}^N l_i}. \tag{1}$$

In the last expression, l_i is the length of the curve γ_i and K is a kernel function similar to those used in nonparametric estimation. A standard choice is the Epanechnikov kernel:

$$K\colon x \mapsto C\left(1 - x^2\right) 1_{[-1,1]}(x),$$

with a normalizing constant C chosen so as to have a unit integral of K on Ω.

Since the entropy is minimal for concentrated distributions, it is quite intuitive to figure out that seeking for a curve system $(\gamma_1, \ldots, \gamma_N)$ giving a minimum value for $E(\gamma_1, \ldots, \gamma_N)$ will induce the following properties:

- The images of the curves tend to get close one to another.
- The individual lengths will be minimized: it is a direct consequence of the fact that the density has a term in γ' within the integral that will favor short trajectories.

Using a standard gradient descent algorithm on the entropy produces an optimally concentrated curve system, suitable for use as a basis for a route network. Figure 2 illustrates this effect on an initial situation given in Figure 1.

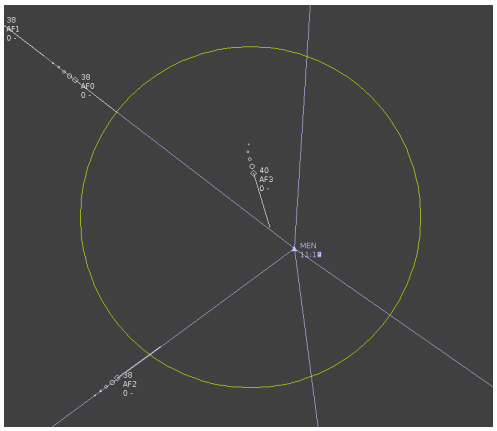

Figure 1. Initial flight plan.

The displacement field for trajectory j is oriented at each point along the normal vector to the trajectory, with norm given by:

$$\int_\Omega \left.\frac{\gamma_j(t) - x}{\|\gamma_j(t) - x\|}\right|_{\mathcal{N}} K'\left(\|\gamma_j(t) - x\|\right) \log \tilde{d}(x) dx \|\gamma_j'(t)\| \tag{2}$$

$$-\left(\int_\Omega K\left(\|\gamma_j(t) - x\|\right) \log \tilde{d}(x)) dx\right) \left.\frac{\gamma_j''(t)}{\|\gamma_j'(t)\|}\right|_{\mathcal{N}} \tag{3}$$

$$+\left(\int_\Omega \tilde{d}(x) \log(\tilde{d}(x)) dx\right) \left.\frac{\gamma_j''(t)}{\|\gamma_j'(t)\|}\right|_{\mathcal{N}}, \tag{4}$$

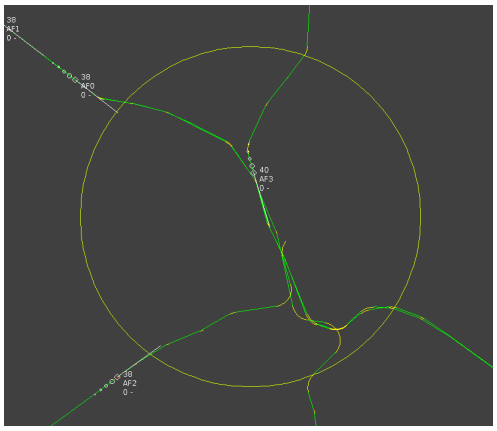

Figure 2. Entropy minimal curve system from the initial flight plan.

where the notation $v_{|\mathcal{N}}$ stands for the projection of the vector v onto the normal vector to the trajectory. An overall scaling constant of:

$$\frac{1}{\sum_{i=1}^{N} l_i},$$

where l_i is the length of trajectory i, has to be put in front of the expression to get the true gradient of the entropy. In practice, it is not needed since algorithms will adjust the size of the step taken in the gradient direction.

V. A LIE GROUP MODELING

While satisfactory in terms of traffic flows, the previous approach suffers from a severe flaw when one considers flight paths that are very similar in shape but are oriented in opposite directions. Since the density is insensitive to direction reversal, flight paths will tend to aggregate while the correct behavior will be to ensure a sufficient separation in order to prevent hazardous encounters. Taking aircraft headings into account in the clustering process is then mandatory when such situations have to be considered.

This issue can be addressed by adding a penalty term to neighboring trajectories with different headings but the important theoretical property of entropy minimization will be lost in the process. A more satisfactory approach will be to take heading information directly into account and to introduce a notion of density based on position and velocity.

Since the aircraft dynamics is governed by a second order equation of motion of the form:

$$\begin{pmatrix} \gamma'(t) \\ \gamma''(t) \end{pmatrix} = F\left(t;\ \begin{matrix} \gamma(t) \\ \gamma'(t) \end{matrix}\right),$$

it is natural to take as state vector:

$$\begin{pmatrix} \gamma(t) \\ \gamma'(t) \end{pmatrix}.$$

The initial state is chosen here to be:

$$\left(\frac{0_d}{e_1}\right),$$

with e_1 the first basis vector, and 0_d the origin in $\mathbb{R}^d$. It is equivalent to model the state as a linear transformation:

$$0_d \otimes e_1 \mapsto T(t) \otimes A(t)(0_d \otimes e_1) = \gamma(t) \otimes \gamma'(t),$$

where $T(t)$ is the translation mapping 0_d to $\gamma(t)$ and $A(t)$ is the composite of a scaling and a rotation mapping e_1 to $\gamma'(t)$. Considering the vector $(\gamma(t), 1)$ instead of $\gamma(t)$ allows a matrix representation of the translation $T(t)$:

$$\left(\frac{\gamma(t)}{1}\right) = \left(\begin{array}{c|c} Id & \gamma(t) \\ \hline 0 & 1 \end{array}\right)\left(\frac{0_d}{1}\right).$$

From now, all points will be implicitly considered as having an extra last coordinate with value 1, so that translations are expressed using matrices. The origin 0_d will thus stand for the vector $(0, \ldots, 0, 1)$ in $\mathbb{R}^{d+1}$. Gathering things yields:

$$\left(\frac{\gamma(t)}{\gamma'(t)}\right) = \left(\begin{array}{c|c} T(t) & 0 \\ \hline 0 & A(t) \end{array}\right)\left(\frac{0_d}{e_1}\right). \tag{5}$$

The previous expression makes it possible to represent a trajectory as a mapping from a time interval to the matrix Lie group $\mathcal{G} = \mathbb{R}^d \times \Sigma \times S\mathbb{O}(d)$, where Σ is the group of multiples of the identity, $S\mathbb{O}(d)$ the group of rotations and $\mathbb{R}^d$ the group of translations. Please note that all the products are direct. The $A(t)$ term in the expression (5) can be written as an element of $\Sigma \otimes S\mathbb{O}(d)$. Starting with the defining property $A(t)e_1 = \gamma'(t)$, one can write $A(t) = \|\gamma'(t)\|U(t)$ with $U(t)$ a rotation mapping $e_1 \in \mathbb{S}^{d-1}$ to the unit vector $\gamma'(t)/\|\gamma'(t)\| \in \mathbb{S}^{d-1}$. For arbitrary dimension d, $U(t)$ is not uniquely defined, as it can be written as a rotation in the plane $\mathcal{P} = \text{span}(e_1, \gamma'(t))$ and a rotation in its orthogonal complement $\mathcal{P}^{\perp}$. A common choice is to let $U(t)$ be the identity in $\mathcal{P}^{\perp}$ which corresponds in fact to a move along a geodesic (great circle) in $\mathbb{S}^{d-1}$. This will be assumed implicitly in the sequel, so that the representation $A(t) = \Lambda(t)U(t)$ with $\Lambda(t) = \|\gamma'(t)\|\text{Id}$ becomes unique.

The Lie algebra $\mathfrak{g}$ of $\mathcal{G}$ is easily seen to be $\mathbb{R}^d \times \mathbb{R} \times \mathbf{Asym}(d)$ with $\mathbf{Asym}(d)$ is the space of skew-symmetric $d \times d$ matrices. An element from $\mathfrak{g}$ is a triple (u, λ, A) with an associated matrix form:

$$M(u, \lambda, A) = \left(\begin{array}{cc|c} 0 & u & 0 \\ 0 & 0 & \\ \hline 0 & & \lambda Id + A \end{array}\right). \tag{6}$$

The exponential mapping from $\mathfrak{g}$ to $\mathcal{G}$ can be obtained in a straightforward manner using the usual matrix exponential:

$$\exp((u, \lambda, A)) = \exp(M(u, \lambda, A)).$$

The matrix representation of $\mathfrak{g}$ may be used to derive a metric:

$$\langle (u, \lambda, A), (v, \mu, B) \rangle_{\mathfrak{g}} = \mathbf{Tr}\left(M(u, \lambda, A)^t M(v, \mu, B)\right).$$

Using routine matrix computations and the fact that A, B being skew-symetric have vanishing trace, it can be expressed as:

$$\langle (u, \lambda, A), (v, \mu, B) \rangle_{\mathfrak{g}} = n\lambda\mu + \langle u, v \rangle + \mathbf{Tr}\left(A^t B\right). \tag{7}$$

A left invariant metric on the tangent space $T_g\mathcal{G}$ at $g \in \mathcal{G}$ is derived from (7) as:

$$\langle\!\langle X, Y, \rangle\!\rangle_g = \langle g^{-1}X, g^{-1}Y \rangle_{\mathfrak{g}},$$

with $X, Y \in T_g\mathcal{G}$. Please note that $\mathcal{G}$ is a matrix group acting linearly so that the mapping g^{-1} is well defined from $T_g\mathcal{G}$ to $\mathfrak{g}$. Using the fact that the metric (7) splits, one can check that geodesics in the group are given by straight segments in $\mathfrak{g}$: if

g_1, g_2 are two elements from $\mathcal{G}$, then the geodesic connecting them is:

$$t \in [0,1] \mapsto g_1 \exp\left(t \log\left(g_1^{-1} g_2\right)\right).$$

where log is a determination of the matrix logarithm. Finally, the geodesic length is used to compute the distance $d(g_1, g_2)$ between two elements g_1, g_2 in $\mathcal{G}$. Assuming that the translation parts of g_1, g_2 are respectively u_1, u_2, the rotations U_1, U_2 and the scalings $\exp(\lambda_1), \exp(\lambda_2)$ then:

$$d(g_1, g_2)^2 = (\lambda_1 - \lambda_2)^2 + \tag{8}$$

$$\mathbf{Tr}\left(\log\left(U_1^t U_2\right) \log\left(U_1^t U_2\right)^t\right) + \|u_1 - u_2\|^2. \tag{9}$$

An important point to note is that the scaling part of an element $g \in \mathcal{G}$ will contribute to the distance by its logarithm.

Based on the above derivation, a flight path γ with state vector $(\gamma(t), \gamma'(t))$ will be modeled in the sequel as a curve with values in the Lie group $\mathcal{G}$:

$$\Gamma : t \in [0,1] \mapsto \Gamma(t) \in \mathcal{G},$$

with:

$$\Gamma(t).(0_d, e_1) = (\gamma(t), \gamma'(t)).$$

In order to make the Lie group representation amenable to statistical thinking, we need to define probability densities on the translation, scaling and rotation components that are invariant under the action of the corresponding factor of $\mathcal{G}$.

VI. Nonparametric estimation on $\mathcal{G}$

Since the translation factor in $\mathcal{G}$ is the additive group $\mathbb{R}^d$, a standard nonparametric kernel estimator can be used. It turns out that it is equivalent to the spatial density estimate of (1), so that no extra work is needed for this component. As for the rotation component, a standard parametrization is obtained recursively starting with the image of the canonical basis of $\mathbb{R}^d$ under the rotation. If R is an arbitrary rotation and $e_1, \ldots, e_d$ is the canonical basis, there is a unique rotation R_{e_1} mapping e_1 to Re_1 and fixing $e_2, \ldots, e_d$. It can be represented by the point $Re_1 = r_1$ on the sphere $\mathbb{S}^{d-1}$. Proceeding the same way for $Re_2, \ldots Re_d$, it is finally possible to completely parametrized R by a $(d-1)$-uple $(r_1, \ldots, r_{d-1})$ where $r_i \in \mathbb{S}^{i-1}$, $i = 1, \ldots, d$. Finding a rotation invariant distribution amounts thus to construct such a distribution on the sphere.

In directional statistics, when we consider the spherical polar coordinates of a random unit vector $u \in \mathbb{S}^{d-1}$, we deal with spherical data (also called circular data or directional data) distributed on the unit sphere. For $d = 3$, a unit vector may be described by means of two random variables θ and φ which respectively represent the co-latitude (the zenith angle) and the longitude (the azimuth angle) of the points on the sphere. Nonparametric procedures, such as the kernel density estimation methods are sometimes convenient to estimate the probability distribution function (p.d.f.) of such kind of data but they require an appropriate choice of kernel functions.

Let $X_1, \ldots, X_n$ be a sequence of random vectors taking values in $\mathbb{R}^d$. The density function f of a random d-vector may be estimated by the kernel density estimator [**?**] as follows:

$$\widehat{f}(x) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{K}_H\left(x - X_i\right), \; x \in \mathbb{R}^d,$$

where $\mathcal{K}_H(x) = \mid H \mid^{-1} \mathcal{K}(H^{-1}x)$, $\mathcal{K}$ denotes a multivariate kernel function and H represents a d-dimensional smoothing matrix, called bandwidth matrix. The kernel function $\mathcal{K}$ is a d-dimensional p.d.f. such as the standard multivariate Gaussian density $\mathcal{K}(x) = (2\pi)^{d/2} \exp\left(-\frac{1}{2} x^T x\right)$ or the multivariate Epanechnikov kernel. The resulting estimation will be the sum of "bumps" above each observation, the observations closed to x giving more important weights to the density estimate. The kernel function $\mathcal{K}$ determines the form of the bumps whereas the bandwidth matrix H determines their width and their orientation. Thereby, bandwidth matrices can be used to adjust for correlation between the components of the data. Usually, an equal bandwidth h in all dimensions is chosen, corresponding to $H = hId$ where Id denotes the $d \times d$ identity matrix. The kernel density estimator then becomes:

$$\widehat{f}(x) = \frac{1}{nh^d} \sum_{i=1}^{n} \mathcal{K}\left(h^{-1}(x - X_i)\right), \; x \in \mathbb{R}^d.$$

In certain cases when the spread of data is different in each coordinate direction, it may be more appropriate to use different bandwidths in each dimension. The bandwidth matrix H is given by the diagonal matrix in which the diagonal entries are the bandwidths $h_1, \ldots, h_d$.

In directional statistics, a kernel density estimate on $\mathbb{S}^{d-1}$ is given by adopting appropriate circular symmetric kernel functions such as von Mises-Fisher, wrapped Gaussian and wrapped Cauchy distributions. A commonly used choice is the von Mises-Fisher (vMF) distribution on $\mathbb{S}^{d-1}$ which is denoted $\mathcal{M}(m, \kappa)$ and given by the following density expression [**?**]:

$$K_{VMF}(x; m, \kappa) = c_d(\kappa)\, e^{\kappa m^T x}, \; \kappa > 0, \; x \in \mathbb{S}^{d-1}, \tag{10}$$

where

$$c_d(\kappa) = \frac{\kappa^{d/2-1}}{(2\pi)^{d/2} I_{d/2-1}(\kappa)} \tag{11}$$

is a normalization constant with $I_r(\kappa)$ denoting the modified Bessel function of the first kind at order r. The vMF kernel function is an unimodal p.d.f. parametrized by the unit mean-direction vector μ and the concentration parameter κ that controls the concentration of the distribution around the mean-direction vector. The vMF distribution may be expressed by means of the spherical polar coordinates of $x \in \mathbb{S}^{d-1}$ [**?**].

Given the random vectors X_i, $i = 1, \ldots, n$, in $\mathbb{S}^{d-1}$, the estimator of the spherical distribution is given by:

$$\widehat{f}(x) = \frac{1}{n} \sum_{i=1}^{n} K_{VMF}(x; X_i)$$

$$= \frac{c_d(\kappa)}{n} \sum_{i=1}^{n} e^{\kappa X_i^T x}, \; \kappa > 0, \; x \in \mathbb{S}^{d-1}.$$

Please, note that the quantity $x - X_i$ which appears in the linear kernel density estimator is replaced by $X_i^T x$ which is the cosine of the angles between x and X_i, so that more important weights are given on observations close to x on the sphere. The concentration parameter κ is a smoothing parameter that plays the role of the inverse of the bandwidth parameter as defined in the linear kernel density estimation. Large values of κ imply greater concentration around the mean direction and lead to undersmoothed estimators whereas small values provide oversmoothed circular densities [**?**]. Indeed, if

$\kappa = 0$, the vMF kernel function reduces to the uniform circular distribution on the hypersphere. Note that the vMF kernel function is convenient when the data is rotationally symmetric.

The vMF kernel function is a convenient choice for our problem because this p.d.f. is invariant under the action on the sphere of the rotation component of the Lie group $\mathcal{G}$. Moreover, this distribution has properties analogous to those of multivariate Gaussian distribution and is the limiting case of a limit central theorem for directional statistics. Other multidimensional distributions might be envisaged, such as the bivariate von Mises, the Bingham or the Kent distributions [**?**]. However, the bivariate von Mises distribution being a product kernel of two univariate von Mises kernels, this is more appropriate for modeling density distributions on the torus and not on the sphere. The Bingham distribution is bimodal and satisfies the antipodal symmetry property $K(x) = K(-x)$. This kernel function is used for estimating the density of axial data and is not appropriate for our clustering approach. Finally, the Kent distribution is a generalization of the vMF distribution, which is used when we want to take into account of the spread of data. However, the rotation-invariance property of the vMF distribution is lost.

As for the scaling component of $\mathcal{G}$, the usual kernel functions such as the Gaussian and the Epanechnikov kernel functions are not suitable for estimating the radial distribution of a random vector in $\mathbb{R}^d$. When distributions are defined over a positive support (here in the case of non-negative data), these kernel functions cause a bias in the boundary regions because they give weights outside the support. An asymmetrical kernel function on $\mathbb{R}^+$ such as the log-normal kernel function is a more convenient choice. Moreover, this p.d.f. is invariant by change of scale. Let $R_1, \ldots, R_n$ be univariate random variables from a p.d.f. which has bounded support on $[0; +\infty[$. The radial density estimator may be defined by means of a sum of log-normal kernel functions as follows:

$$\widehat{g}(r) = \frac{1}{n} \sum_{i=1}^{n} K_{LN}(r; \ln R_i, h), r \geq 0, h > 0,$$

where

$$K_{LN}(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-\frac{(\ln x - \mu)^2}{2\sigma^2}}$$

is the log-normal kernel function and h is the bandwidth parameter. The resulting estimate is the sum of bumps defined by log-normal kernels with medians R_i and variances $(e^{h^2} - 1)e^{h^2} R_i^2$. Note that the log-normal (asymmetric) kernel density estimation is similar to the kernel density estimation based on a log-transformation of the data with the Gaussian kernel function. Although the scale-change component of $\mathcal{G}$ is the multiplicative group $\mathbb{R}^+$, we can use the standard Gaussian kernel estimator and the metric on $\mathbb{R}$.

VII. Unsupervised entropy clustering

The first thing to be considered is the extension of the entropy definition to curve systems with values in $\mathcal{G}$. Starting with expression from (1), the most important point is the choice of the kernel involved in the computation. As the group $\mathcal{G}$ is a direct product, choosing $K = K_t.K_s.K_o$ with K_t, K_s, K_o functions on respectively the translation, scaling and rotation part will yield a $\mathcal{G}$-invariant kernel provided the $K_t, K_s K_o$ are invariant on their respective components. Since the translation part of $\mathcal{G}$ is modeled after $\mathbb{R}^n$, the epanechnikov kernel is a suitable choice. As for the scaling and rotation, the choice made follows the conclusion of section VI: a log-normal kernel and a von-Mises one will be used respectively. Finally, the term $\|\gamma'(t)\|$ in the original expression of the density, that is required to ensure invariance under re-parametrization of the curve, has to be changed according to the metric in $\mathcal{G}$ and is replaced by $\langle\!\langle \gamma'(t), \gamma'(t) \rangle\!\rangle_{\gamma(t)}^{1/2}$. The density at $x \in \mathcal{G}$ is thus:

$$d_{\mathcal{G}}(x)) = \frac{\sum_{i=1}^{N} \int_0^1 K\left(x, \gamma_i(t)\right) \langle\!\langle \gamma_i'(t), \gamma_i'(t) \rangle\!\rangle_{\gamma_i(t)}^{1/2} dt}{\sum_{i=1}^{N} l_i} \quad (12)$$

where l_i is the length of the curve in $\mathcal{G}$, that is:

$$l_i = \int_0^1 \langle\!\langle \gamma_i'(t), \gamma_i'(t) \rangle\!\rangle_{\gamma_i(t)}^{1/2} dt \quad (13)$$

The expression of the kernel evaluation $K\left(x, \gamma_i(t)\right)$ is split into three terms. In order to ease the writing, a point x in $\mathcal{G}$ will be split into x^r, x^s, x^o components where the exponent r, s, t stands respectively for translation, scaling and rotation. Given the fact that K is a product of component-wise independent kernels it comes:

$$K\left(x, \gamma_i(t)\right) = K_t\left(x^t, \gamma_i^t(t)\right) K_s\left(x^s, \gamma_i^s(t)\right) K_o\left(x^o, \gamma_i^o(t)\right)$$

where:

$$K_t(x^t, \gamma_i^t(t)) = \text{ep}\left(\|x^t - \gamma_i^t(t)\|\right) \quad (14)$$

$$K_s(x^s, \gamma_i^s(t)) = \frac{1}{x^s \sigma \sqrt{2\pi}} \exp\left(-\frac{\left(\log x^s - \log \gamma_i^s(t)\right)^2}{2\sigma^2}\right) \quad (15)$$

$$K_o(x^o, \gamma_i^o(t)) = C(\kappa) \exp\left(\kappa \mathbf{Tr}\left(x^{ot} \gamma_i^o(t)\right)\right) \quad (16)$$

with $C(\kappa)$ the normalizing constant making the kernel of unit integral. Please note that the expression given here is valid for arbitrary rotations, but for the application targeted by the work presented here, it boils down to a standard von-mises distributions on $\mathbb{S}^{d-1}$:

$$K_o(x^o, \gamma_i^o(t)) = C(\kappa) \exp\left(\kappa x^{ot} \gamma_i^o(t)\right)$$

with normalizing constant as given in (11). In the general case, it is also possible, writing the rotation as a sequence of moves on spheres $\mathbb{S}^{d-1}, \mathbb{S}^{d-2}, \ldots$ and the distribution as a product of von-Mises on each of them, to have a vector of parameters κ: it is the approach taken in [**?**] and it may be applied verbatim here if needed.

The entropy of the system of curves is obtained from the density in $\mathcal{G}$:

$$E(d_{\mathcal{G}}) = -\int_{\mathcal{G}} d_{\mathcal{G}}(x) \log d_{\mathcal{G}}(x) d\mu_{\mathcal{G}}(x) \quad (17)$$

with $d\mu_{\mathcal{G}}$ the left Haar measure. Using again the fact that $\mathcal{G}$ is a direct product group, $d\mu$ is easily seen to be a product measure, with dx^t, the usual Lebesgue measure on the translation part, dx^s/x^s on the scaling part and the lebesgue measure dx^o on $\mathbb{S}^{d-1}$ for the rotation part. It turns out that the $1/x^s$ term in the expression of dx^s/x^s is already taken into account in the kernel definition, due to the fact that it is expressed

in logarithmic coordinates. The same is true for the Von-Mises kernel, so that in the sequel only the (product) lebesgue measure will appear in the integrals.

Finding the system of curves with minimum entropy requires a displacement field computation as detailed in [**?**]. For each curve γ_i, such a field is a mapping $\eta_i \colon [0,1] \to T\mathcal{G}$ where at each $t \in [0,1]$, $\eta_i(t) \in T\mathcal{G}_{\gamma_i(t)}$.Compare to the original situation where only spatial density was considered, the computation must now be conducted in the tangent space to $\mathcal{G}$. Even for small problems, the effort needed becomes prohibitive. The structure of the kernel involved in the density can help in cutting the overall computations needed. Since it is a product, and the translation part is compactly supported, being an epanechnikov kernel, one can restrict the evaluation to points belonging to its support. Density computation will thus be made only in tubes around the trajectories.

Second, for the target application that is to cluster the flight paths into a route network and is of pure spatial nature, there is no point in updating the rotation and scaling part when performing the moves: only the translation part must change, the other two being computed from the trajectory. The initial optimization problem in $\mathcal{G}$ may thus be greatly simplified.

Let ϵ be an admissible variation of curve γ_i, that is a smooth mapping from $[0,1]$ to $T\mathcal{G}$ with $\epsilon(t) \in T_{\gamma_i(t)}\mathcal{G}$ and $\epsilon(0) = \epsilon(1) = 0$. We assume furthermore that ϵ has only a translation component. The derivative of the entropy $E(d_\mathcal{G})$ the t curve γ_i is obtained from the first order term when γ_i is replaced by $\gamma_i + \epsilon$. First of all, it has to be noted that $d_\mathcal{G}$ is a density and thus has unit integral regardless of the curve system. When computing the derivative of $E(d_\mathcal{G})$, the term

$$-\int_\mathcal{G} d_\mathcal{G}(x)\frac{\partial_{\gamma_i} d_\mathcal{G}(x)}{d_\mathcal{G}(x)} d\mu_\mathcal{G}(x) = -\int_\mathcal{G} \partial_{\gamma_i} d_\mathcal{G}(x) d\mu_\mathcal{G}(x)$$

will thus vanish. It remains:

$$-\int_\mathcal{G} \partial_{\gamma_i} d_\mathcal{G}(x) \log d_\mathcal{G}(x) d\mu_\mathcal{G}(x)$$

The density $d_\mathcal{G}$ is a sum on the curves, and only the i-th term has to be considered. Starting with the expression from (12), one term in the derivative will come from the denominator. It computes the same way as in [**?**] to yield:

$$\left.\frac{\gamma_i^{t\prime\prime}(t)}{\langle\!\langle \gamma_i'(t), \gamma_i'(t)\rangle\!\rangle_\mathcal{G}}\right|_\mathcal{N} E(d_\mathcal{G}) \tag{18}$$

Please note that the second derivative of γ_i is considered only on its translation component, but the first derivative makes use of the complete expression. As before, the notation $|_\mathcal{N}$ stands for the projection onto the normal component to the curve.

The second term comes from the variation of the numerator. Using the fact that the kernel is a product $K^t K^s K^o$ and that all individual terms have a unit integral on their respective components, the expression becomes very similar to the case of spatial density only and is:

$$-\left(\int_\mathcal{G} K\left(x, \gamma_i(t)\right) \log d_\mathcal{G}(x) d\mu_{\mathcal{G}(x)}\right) \left.\frac{\gamma_i^{t\prime\prime}(t)}{\langle\!\langle \gamma_i'(t), \gamma_i'(t)\rangle\!\rangle_\mathcal{G}^{1/2}}\right|_\mathcal{N} \tag{19}$$

$$+\int_{\mathbb{R}^d} e(t) K^{t\prime}\left(x^t, \gamma_i^t(t)\right) \log d_\mathcal{G}(x) \langle\!\langle \gamma_i'(t), \gamma_i'(t)\rangle\!\rangle_\mathcal{G}^{1/2} dx^t \tag{20}$$

with:

$$e(t) = \left.\frac{\gamma_i^t(t) - x^t}{\|\gamma_i^t(t) - x^t\|}\right|_\mathcal{N}$$

VIII. Results

Only partial results are available for the moment and several traffic situations are still to be considered. On simple synthetic examples, the algorithm works as expected, avoiding going to close to trajectories with opposite directions as indicated on Figure 3.

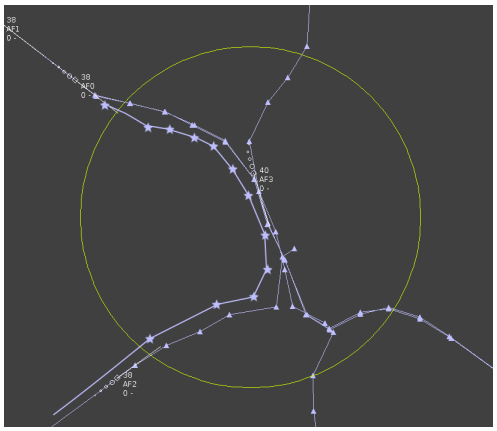

Figure 3. Clustering using the Lie approach

In a more realistic setting, the arrivals and departures at Toulouse Blagnac airport were analyzed. The algorithm performs well as indicated on Figure 4. Four clusters are identified, with mean lines represented through a spline smoothing between landmarks. It is quite remarkable that all density based algorithms were unable to separate the two clusters located at the right side of the picture, while the present one clearly show a standard approach procedure and a short departure one.

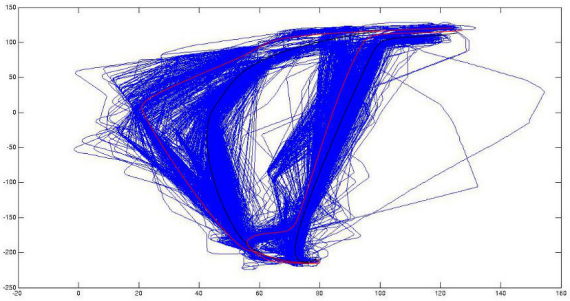

Figure 4. Bundling trajectories at Toulouse airport

An important issue still to be addressed with the extended algorithm is the increase in computation time that reaches 20 times compared to the appoach using only spatial density entropy. In the current implementation, the time needed to cluster the traffic presented in Figure 3 is in the order of 0.01s on a XEON 3Ghz machine and with a pure java implementation. For the case of Figure 4, 5 minutes are needed on the same machine for dealing with the set of 1784 trajectories.

IX. Conclusion and future work

The entropy associated with a system of curves has proved itself efficient in unsupervised clustering application where shape constraints must be taken into account. For using it in aircraft route design, heading and velocity information must be added to the state vector, inducing an extra level of complexity. The present work relies on a Lie group modeling as an unifying approach to state representation. It has successfully extended the notion of curve system entropy to this setting, allowing the heading/velocity to be added in a intrinsic way. The method seems promising, as indicated by the results obtained on simple synthetic situations, but extra work needs to be dedicated to algorithmic efficiency in order to deal with the operational traffic datasets, in the order of tens of thousand of trajectories.

Generally speaking, introducing a Lie group approach to data description paves the way to new algorithms dedicated to data with a high level of internal structuring. Studies are initiated to address several issues in high dimensional data analysis using this framework.

References