Machine Learning with Applications 11 (2023) 100446

Contents lists available at ScienceDirect

Machine Learning with Applications

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

Deep generative modelling of aircraft trajectories in terminal maneuvering areas

Timothé Krauth [a,b,*], Adrien Lafage [b], Jérôme Morio [b], Xavier Olive [b], Manuel Waltert [a]

[a] Zurich University of Applied Sciences, Centre for Aviation, Winterthur, Switzerland
[b] ONERA/DTIS, Université de Toulouse, F-31055 Toulouse, France

ARTICLE INFO

Keywords:
Air traffic management
Deep Generative Models
Multivariate time-series generation
Variational Autoencoder
Temporal Convolutional Networks

ABSTRACT

Airspace design is subject to a multitude of constraints, which are mainly driven by the concern to keep the risk of mid-air collision below a target level of safety. For that purpose, Monte Carlo simulation methods can be applied to estimate aircraft conflict probability but require the accurate generation of artificial trajectories. Generative models allow to generate an infinite number of trajectories for air traffic procedures where only few observations are available. The generated trajectories must not only resemble observed trajectories in terms of statistical distributions but they should stay flyable and consider uncertainty due to weather, air traffic control, aircraft performances, or human factors. This paper focuses on the generation problem, and its main contribution lies in the adaptation of the Variational Autoencoder structure to the problem of 4-dimensional aircraft trajectories modelling using Temporal Convolutional Networks and a prior distribution composed of a Variational Mixture of Posteriors (VampPrior). The proposed model has been trained on trajectories in the Terminal Manoeuvre Area of Zurich airport, which have a particularly high degree of variability as air traffic controllers often take actions that deviate aircraft from the nominal approach procedure. The model has demonstrated great abilities to take into account such amount of uncertainty. Regarding metrics that evaluate the estimation of the statistical distribution of the observed trajectories, and the flyability of the generated ones, the proposed method outperforms traditional statistical methods by being able to generate more complex and realistic trajectories.

1. Introduction

Although the airspace over Switzerland is very busy because of its central location in Europe, it has been built up over many decades through a series of ad-hoc and disparate modifications. This system of modifications has reached its limits and is increasingly struggling to cope with the current needs of air traffic. The Federal Office of Civil Aviation set up in 2016 the AVISTRAT-CH program,[1] which aims to address the issue with a more modern design of the Swiss Airspace, without compromising the safety level. Air traffic controllers have to maintain specific horizontal and vertical separation distances between aircraft. The violation of these minimum distances is called *loss of separation* (LoS) and is considered safety critical. Consequently, it is essential to have efficient *collision risk models* (CRM) to accurately monitor the probability of LoS occurrences to ensure that they are below the target level of safety, which specifies the socially accepted level of safety.

The estimation of the probability of LoS events may be carried out based on Monte Carlo methods. However, LoS occurrences are extremely rare, and thus a large number of simulations of pairs of trajectories must be conducted to observe only few of them. To be consistent, the set of simulations must be conducted on a large set of trajectories in order to be able to observe a sufficiently high number of successes. But the number of observed trajectories for given conditions is limited. For example, in order to collect 1 million landing trajectories at Zurich airport, we would have to download about 10 years of data. However, apart from the download and processing time, using such old data to estimate the current probability of LoS events is not relevant because air traffic procedures and aircraft types have changed. Ideally, we would like to work on 1 million trajectories that have

* Correspondence to: Technikumstrasse 71, 8400 Winterthur, Switzerland.
E-mail addresses: timothe.krauth@zhaw.ch (T. Krauth), adrien.lafage@ensta-paris.fr (A. Lafage), jerome.morio@onera.fr (J. Morio), xavier.olive@onera.fr (X. Olive), manuel.waltert@zhaw.ch (M. Waltert).

[1] https://www.bazl.admin.ch/bazl/en/home/themen/luftfahrtpolitik/avistrat.html.

https://doi.org/10.1016/j.mlwa.2022.100446
Received 19 October 2022; Received in revised form 22 December 2022; Accepted 22 December 2022
Available online 28 December 2022

Abbreviations

ADS-B	Automatic Dependent Surveillance -Broadcast
AE	Autoencoder
ASMA	Arrival Sequencing and Metering Area
ATC	Air Traffic Control
CNN	Convolutional Neural Network
CRM	Collision Risks Model
DGM	Deep Generative Model
DTW	Dynamic Time Wrapping
ELBO	Evidence Lower BOund
FCN	Fully Connected Network
FCVAE	Fully Connected Variational Autoencoder
ft	feet
GAN	Generative Adversarial Network
GMM	Gaussian Mixtures Models
KL-divergence	Kullback–Leibler Divergence
kts	knots
LoS	Loss of separation
LSZH	Identifier for Zurich airport
MFA	Mean Field Assumption
nm	nautical miles
PCA	Principal Component Analysis
RNN	Recurrent Neural Network
SSPD	Symmetric Segment-Path Distance
STAR	Standard Terminal Arrival Route
TCN	Temporal Convolutional Network
TMA	Terminal Maneuvering Area
TCVAE	Temporal Convolutional Variational Autoencoder
VAE	Variational Autoencoder
VampPrior	Variational Mixture of Posteriors

been observed recently, under similar conditions. Generative models allow generating an arbitrary large amount of trajectories based on the most recent observations, or even procedures that are rarely used, but potentially dangerous. Consequently, previous research such as Eckstein (2010), Henry, Schmitz, Revenko, and Kelbaugh (2013) and Jacquemart and Morio (2013) developed different analytic generators of random aircraft trajectories to be used in conjunction with Monte Carlo simulations. Nevertheless, the generation methods in the literature are often limited as they only generate simple or one-dimensional trajectories, which confines the collision risk analysis to sub-problems. For instance, Jacquemart and Morio (2013) generate trajectories to estimate collisions between en-route trajectories with constant altitude and heading. Henry et al. (2013) create 1-dimensional altitude profiles for the study of airborne collisions between intersecting runways. The objective of this paper is to produce a more general approach by developing a generative process for synthetic 4-dimensional complex trajectories for Terminal Maneuvering Area (TMA), i.e. the controlled airspace surrounding a major airport where there is a high volume of traffic. Traffic in TMA is particularly complex and diversified, as air traffic controllers often take actions that deviate aircraft from the nominal approach procedure. We have defined four requirements that the proposed method must meet: (i) real and synthetic trajectories should share the same statistical distribution, (ii) synthetic trajectories should look realistic regarding the law of physics, (iii) the generation should provide a wide diversity of trajectories, including some that might never be observed from an operational point of view, (iv) the user should be able to select the general shape of the generated random synthetic trajectories. We hope to be able to orient the generation process towards specific types of trajectories (e.g. the random generation of approaches from the South with holding patterns).

To generate aircraft trajectories, the literature mentions both model-driven and data-driven approaches. Model-driven methods are based on flight mechanic equations and emphasize the physical reality of the generated trajectories (requirement ii) (Delahaye, Puechmorel, Tsiotras, & Féron, 2014). However, introducing randomness into these deterministic models is complicated, and it is difficult to capture the full amount of variability present in complex flight patterns (requirements i and iii). *Data-driven models* (Krauth, Morio, Olive, Figuet, & Monstein, 2021) mimic the distribution of observed trajectories to produce synthetic trajectories that are identically distributed (requirement i). Therefore, they are very effective in capturing the high uncertainty present in the trajectories (iii), but generation may lack physical realism (requirement ii). In the context of Monte Carlo simulations, where synthetic trajectories must render the uncertainty contained in observed trajectories, generative data-driven models appear the most suitable. They allow drawing an arbitrary large amount of random synthetic trajectories, without having to specify initial conditions, flight parameters, or atmospheric scenarios. To the best of the authors' knowledge, previous research on data-driven trajectory generation focuses exclusively on basic statistical density estimation methods (Murça & de Oliveira, 2020), and/or confines itself to rather simple case studies (Eckstein, 2010; Henry et al., 2013). Currently, existing generative data-driven methods struggle to produce 4-dimensional synthetic trajectories (latitude, longitude, altitude, time) for complex flight operations, such as full approach procedures taking into account air traffic controller actions. While Murça and de Oliveira (2020) address the issue with a similar approach, they model approach trajectories focusing more on their general shape than the specificity of each (shortcuts, loops, additional heading changes, etc.). We take the idea further by better taking into account trajectories that do not follow standard approach procedures, which guarantees a greater diversity in the generated trajectories.

The main contribution of this paper lies in the adaptation of the Variational Autoencoder (VAE) by Kingma and Welling (2013) to the modelling of multivariate time-series with high temporal dependencies. Unlike model-driven approaches, the proposed data-driven method estimate the distribution of the underlying data and then automatically takes into account sources of uncertainty as it mimics the variability observed in approaching trajectories (requirements i and iii). It can generate random trajectories for different aircraft types, incoming from all possible standard terminal arrival routes (STARs), and influenced by controllers' actions which deviate flights from the standard procedure. Additionally, this paper shows that the use of VAE improves significantly the estimation of the distribution of aircraft trajectories over classical multivariate density estimation methods such as Gaussian Mixtures (requirement i). The generated trajectories are then statistically closer to the observed ones (requirement ii). Similar to Murça and de Oliveira (2020) who generate random trajectories around what the algorithm identifies as the main approach routes, our method generates random trajectories around *pseudo-inputs*: artificially created trajectories that cover the distribution of observed trajectories, which allow selecting the form of trajectories to be generated (requirement iv). Finally, once the VAE trained, the generation of new trajectories is instantaneous. The implementation can be found on GitHub.[2]

The remainder of this paper is structured as follows: the literature review in Section 2 highlights the main generation methods for aircraft trajectories. Section 3 presents the case study in which the generation method has been developed. Section 4 describes the VAE framework and improvements that have been made to deal with multivariate time series. The results of Section 5 are divided into three parts. Section 5.1 highlights how the proposed VAE architecture improves the learning

[2] https://github.com/kruuZHAW/deep-traffic-generation-paper.

efficiency for the problem of trajectory generation over classical VAE architectures. Then, Section 5.2 analyses the quality of the generated trajectories in the light of the goodness-of-fit with the underlying distribution of observed trajectories, and the realistic nature of the trajectories generated. It compares the proposed VAE architecture with the data-driven generation method from Murça and de Oliveira (2020). Finally, Section 5.3 exposes the generation process of the proposed method. To conclude, a conclusion and future works are given in Section 6.

2. Literature review

The literature presents a wide variety of trajectory generation methods, the efficiency of which depends on the objective to be achieved. The bulk of the literature focuses on model-driven methods that often consist of generating one aircraft trajectory that follows the flight dynamic equations, while optimizing a given criterion, usually fuel consumption or flight time. It can also be constrained to avoid collisions with static or moving obstacles (Delahaye et al., 2014; Koyuncu, Uzun, & Inalhan, 2016). However, each generated trajectory requires precise knowledge of a multitude of input parameters, such as aircraft performance, or flight and environmental conditions, which makes model-driven methods badly designed to capture uncertainty, and to generate large random sets of diverse artificial trajectories, as stated by Henry et al. (2013, p.4). In model-driven methods, randomness can be achieved by including variability in the inputs and/or the outputs. But as sources of uncertainty are often not observed, the randomness introduced may be significantly different from that observed in reality. This literature review first presents existing data-driven generation methods for aircraft trajectories. Then, it highlights what are the current state-of-the-art generation methods in Deep Learning called Deep Generative Models (DGM). Eventually, tools to evaluate the generated trajectories are introduced.

Model-driven methods are well suited to determine trajectories under specific constraints but rather inefficient to take into account uncertainty, whereas data-driven models are to be preferred. The latter are based on the estimation of complicated statistical distributions to mimic the information contained in the observed data. Jacquemart and Morio (2013) deduce from real observations a stochastic process for en-route aircraft trajectories to be used with advanced Monte Carlo simulation schemes. The stochasticity represents the deviation from the initial line due to wind, tracking, navigation or control. However, it can only be applied to simple 2-dimensional scenarios of LoS probability estimations (Jacquemart & Morio, 2016) between straight trajectories with constant altitude and speed. Murça and de Oliveira (2020) cluster the approaching trajectories in São Paulo Airport to identify the main trajectory patterns. Each of them is then modelled with a component from a Gaussian Mixture distribution. Sampling in one of those components generates artificial trajectories with a shape matching the corresponding operational pattern. Dimensionality reduction is also a tool designed to improve the estimation of complex statistical distributions and reduce the impact of the curse of dimension. For instance, Eckstein (2010) reduces the dimension of observed ground speed profiles with a Principal Component Analysis (PCA) and generates synthetic profiles by sampling new points in the latent space, before serving as input to the inverse linear operation. The same idea is presented in Henry et al. (2013) and applied to collision risk modelling for converging runways. However, both articles only allow to consider one particular dimension (altitude or ground speed) of straight trajectories. Moreover, PCA is a useful tool to project observed trajectories in a smaller space of representation, but there are no mathematical guarantees that a newly sampled point in the latent space will correspond to a relevant trajectory once decoded. Jarry, Hassoumi, Delahaye, and Hurter (2020) extend the dimensionality reduction approach by considering trajectories as continuous objects and using functional PCA, whereas the projection in the latent space allows to better identify pattern similarities. The authors show that resorting to dimensionality reduction is a promising idea to deal with 4-dimensional trajectories, but the method only allows to modify existing patterns, and not to perform a proper generation, as PCA is not designed to decode points from the latent space that do not correspond to observed trajectories. Krauth et al. (2021) apply dimensionality reduction for the sole purpose of improving the probability density estimate of the observed trajectories. The trajectories are not represented by their latitude and longitude anymore, but by their projection on fixed perpendicular lines. The operator is deterministic, but only allows to consider 2-dimensional paths with simple patterns. Lazzara et al. (2022) and Zhang, Hu, and Du (2022) use autoencoders as a non-linear projection operator to extract information from high-dimensional time-series in order to facilitate their analysis. Jarry, Couellan, and Delahaye (2019) go further into the complexity of the generation method by using Generative Adversarial Networks (GAN), which are capable of reconstructing an aircraft trajectory from a random vector of a smaller dimension. However, the latent space here does not give any insights into the organization of the observed trajectories, and the method was only tested on very simple trajectory patterns.

Each method outlined above contains promising ideas in terms of dimensionality reduction and probability density estimation. However, most trajectory generation models use classical methods of estimation, which do not seem sufficiently developed to handle distributions of complex 4-dimensional trajectories, with patterns as complex as those found in TMAs. Finding ways to improve the goodness-of-fit is a core question, and more powerful density estimation models seem to be required. Deep Generative Models (DGM) are neural networks that are trained to approximate complicated and high-dimensional probability distributions to describe the way the underlying data has been generated. They represent currently one of the most important field of research in deep learning, and consequently benefit from active and recent studies; whether it is for the generation of images, videos (Vondrick, Pirsiavash, & Torralba, 2016) or even sensitive data such as medical data (Lenz, Hess, & Binder, 2021). Different frameworks exist, and Jarry et al. (2019) already explored the use of Generative Adversarial Networks (GAN) (Goodfellow et al., 2014). In this paper, we focus on Variational Autoencoders (VAE) (Kingma & Welling, 2013). VAE are directly in line with the trajectory generation methods presented previously. They perform dimensionality reduction thanks to their encoder/decoder architecture, but also provide a mathematically justified framework for the generation. Moreover, compared to the GAN from Jarry et al. (2019), VAE are more stable to train, and benefit from a better explainability.

Evaluating the quality of the generated trajectories is of utmost importance, but the criterion of evaluation may depend on the objective of the generation model. For instance, model-driven methods are designed to generate trajectories that follow the flight dynamic equations. It is therefore relevant to evaluate them on their ability to follow the distribution of observed trajectories. It is the opposite for data-driven methods, and it will be more suitable to evaluate them on their ability to produce physically realistic trajectories. Olive, Sun, Murca, and Krauth (2021) tackle the problem of the evaluation of generated trajectories and propose suitable metrics for each type of generation method. In particular, the authors have developed a method for evaluating data-driven models, based on a trajectory simulator, here BlueSky (Hoekstra & Ellerbroek, 2016).

3. Problem and data

In this paper, we focus on air traffic in the TMA of Zurich airport. To guarantee optimal smooth operations throughout the year and in all climatic conditions, while respecting the political and environmental constraints of the surrounding areas, the airport employs three operating concepts.[3] described in Fig. 1 The choice of the concept depends

[3] https://www.flughafen-zuerich.ch/en/company/media-policy-and-investors/politics-and-business/operating-concepts.

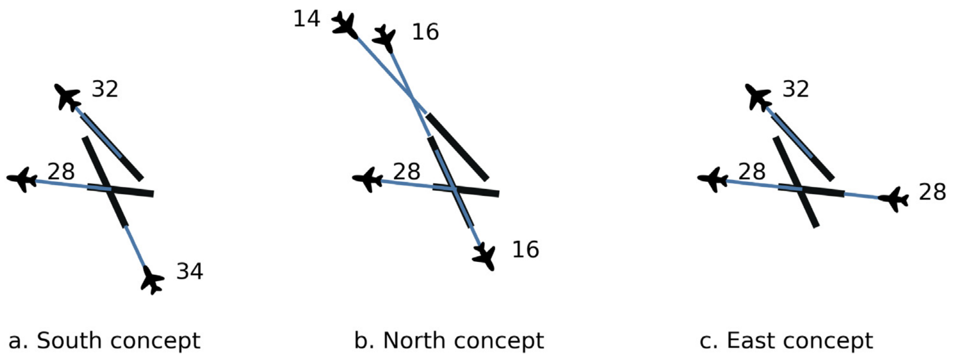

Fig. 1. Operating concepts at Zurich Airport.

mainly on the time of day. As such, the North Operating Concept is applied most, as it is usually active during the day. In this operating concept, aircraft land on runway 14. Landing trajectories are significantly more complex than take-off trajectories, as they can approach from all directions, and air traffic controllers heavily influence traffic to achieve an appropriate landing sequence. Consequently, approaching trajectories might contain complex procedures that cannot be observed in other situations, such as *holding patterns*, i.e. racetrack shapes for stacking aircraft, or a significant number of heading changes in a short time (see Fig. 2). Approach procedures in Zurich Airport are very representative of the pool of scenarios observable at most of the world's major airports, with operational constraints due to the surrounding terrain, noise abatement, or emission mitigation. The efficiency of a given algorithm to generate synthetic trajectories based on data observed at Zurich airport is then expected to be representative of its performance on other airport data. Moreover, it is also expected to give satisfying results for simpler patterns, such as en-route trajectories or departures.

A total of 14,000 landing trajectories on runway 14 in Zurich Airport (LSZH) were collected through Automatic Dependent Surveillance-Broadcast (ADS-B) data from the OpenSky Network (Schäfer, Strohmeier, Lenders, Martinovic, & Wilhelm, 2014) between 1 October and 30 November 2019. They are directly available in the traffic library (Olive, 2019) in Python. Trajectories are trimmed within 40 nautical miles from Zurich airport (Arrival Sequencing and Metering Area, ASMA) and end 1.5 nm after the Final Approach Point of runway 14. Each trajectory is then modified in such a way that they contain exactly 200 data points by using linear interpolation in order to have a representation of trajectories which is smooth enough. In this way, trajectories have a consistent representation and data points are close enough to prevent high gradients from one point to another. Approaches leading to go-arounds were excluded because very few were actually observed. The generation of go-arounds would require to build a dedicated dataset on several years of observation. This was done previously in Krauth et al. (2021). The sample used for the study is represented on Fig. 2. For each point of the trajectory, values for the track (angle), the ground speed, the altitude, and the cumulative time from the entry point are kept. We have selected track instead of latitude and longitude because it produces smoother trajectories. The model does not have to learn complicated correlations between the latitude and the longitude. As trajectories are processed in order to end at the same point, it is possible to retrieve the latitude and longitude for each timestamp thanks to the track and the ground speed. As a result, one trajectory is described by a matrix in $\mathbb{R}^{4\times200}$.

4. Methodology

Generative modelling involves the estimation of the joint distribution over all the variables to mimic the generation process of observed data. For aircraft trajectories described by $(\text{track}_i, \text{groundspeed}_i, \text{altitude}_i, \text{time}_i)$ for i in $[0, \dots n]$ with $n = 200$ observations, the model has to estimate a distribution in dimension 800, which cannot be

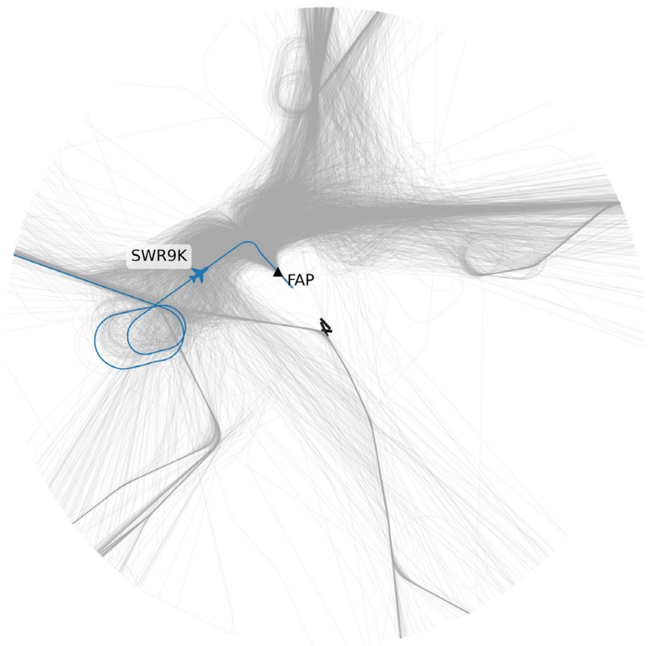

Fig. 2. Historical landing trajectories downloaded with OpenSky Network between 1st October and 30 November at Zurich Airport for runway 14.

done with classical statistical methods such as marginal-copula decomposition, as applied in Krauth et al. (2021). When it comes to the estimation of complex multivariate probability densities, resorting to dimensionality reduction is often a good practice, as a high number of features often leads to weaker goodness of fit due to the *Curse of Dimensionality*. However, dimensionality reduction techniques such as PCA (Eckstein, 2010; Henry et al., 2013) or Autoencoders (Olive, Basora, Viry, & Alligier, 2020) are used to represent observed data, but do not offer the guarantee that a randomly drawn point in the latent space distribution will be meaningful when decoded. The latent space should be endowed with the required properties for generation, namely *continuity* and *completeness*. The former means that two close points in the latent space should look alike once decoded, whereas the latter states that a point sampled within the latent space distribution should be meaningful once decoded. Variational Autoencoders (VAE) are directly based on these concepts, and have the advantage of: (i) being capable of estimating multivariate distributions that are far more complex than traditional statistical methods, (ii) being fairly stable to train compared to generative adversarial networks, and (iii) giving the explicit generative distribution in the latent space and thus enabling the setup of Monte Carlo simulations. Overall, VAEs can be summarized by two main principles. First, they encode data into a smaller dimensional latent space, and second, they regularize the distribution in this latent space to ensure new samples can be generated.

4.1. Variational autoencoder framework

VAE was first introduced by Kingma and Welling (2013) and extensively explained in Kingma and Welling (2019). As the name suggests, it has been built upon the framework of Bayesian Variational Inference, which uses a set of unobserved latent variables z to facilitate the estimation of the distribution of the initial data $p^*(x)$. We will use θ as a notation for the parameters of the models (such as neural network weights). p_θ is then the model parametrized by θ to estimate the real distribution p^*.

$$p_\theta(x) = \int p_\theta(x,z)\,dz \tag{1}$$

The marginalization over an unknown set of variables makes the computation of the integral intractable (the search space of z can be combinatorially large for instance). Nevertheless, the Bayes theorem yields to:

$$p_\theta(z \mid x) = \frac{p_\theta(x,z)}{p_\theta(x)} = \frac{p(z)p_\theta(x \mid z)}{\int p_\theta(x,z)\,dz} \tag{2}$$

Indeed, $p_\theta(x,z)$ is efficient to compute. Therefore, being able to infer $p_\theta(z \mid x)$ enables the computation of $p_\theta(x)$. Variational Inference is the statistical method that estimates $p_\theta(z \mid x)$ by using an approximate $q_\phi(z \mid x)$ selected within a chosen family of distributions. The density $q_\phi(z \mid x)$ is called the variational posterior distribution, $p_\theta(z \mid x)$ the true posterior, and $p(z)$ the prior. As the Gaussian distribution family are often considered in VAEs, the density of a Gaussian distribution $\mathcal{N}(\mu_x, \Gamma_x)$ of mean μ_x and covariance matrix Γ_x is denoted $f(\cdot \mid \mu_x, \Gamma_x)$ in the following. VAEs leverage the use of a Gaussian variational posterior distribution to find a good estimate of $p_\theta(x)$ through three components:

- *the probabilistic encoder* (or *inference network*) performs dimensionality reduction by mapping an input trajectory x into parameters for the posterior distribution in smaller dimension M. Most of the time, a Gaussian distribution $\mathcal{N}(\mu_x, \Gamma_x)$ is considered for the posterior distribution. Thus, $(\mu_x, \Gamma_x) = E_\phi(x)$, where E_ϕ is the encoder neural network, and $q_\phi(z|x) = f(z \mid \mu_x, \Gamma_x)$. One has to be aware that each input data x is associated with a Gaussian distribution with a different set of parameters (μ_x, Γ_x). Variational Inference frameworks often rely on the Mean Field Assumption (MFA) that states that the posterior distribution can be factorized. Even it is not mandatory, it accelerates training by reducing the estimation of the covariance matrix of the posterior to its diagonal. It also makes the dimensions of the latent space independent, which enables the detection of the most important dimensions in the latent representation (Asperti & Trentin, 2020).
- *the latent space* represents the space in smaller dimension in which inputs are projected, and is distributed according to the aggregated approximate posterior distribution:

$$\frac{1}{\mathrm{Card}(\mathbb{X})} \sum_{x \in \mathbb{X}} q_\phi(z \mid x), \tag{3}$$

where $\mathbb{X}$ is the training dataset and $\mathrm{Card}(\mathbb{X})$ its cardinality. The training loss of the VAE encourages all approximate posteriors $q_\phi(z \mid x)$ to be close to the prior $p(z)$. In a perfect world, the approximate posterior $q_\phi(z \mid x)$ matches both the real posterior $p(z \mid x)$ and the prior $p(z)$. In this case, the Bayes' rule states that $p_\theta(x) = p_\theta(x \mid z)$, which is exactly what is expected from a generative model, namely being able to deduce exactly the unknown distribution of x from the one of z. Sampling a point from the aggregated posterior leads to the reconstruction of an existing trajectory. Alternatively, sampling a point from the prior leads to generating a new trajectory. The larger the dimension M of the latent space is, the less information is lost during encoding, but also, the more difficult it is to match the aggregated posterior and the prior.
- *the probabilistic decoder* (or *generative network*) maps one point sampled from $f(z \mid \mu_x, \Gamma_x)$ for a given x into parameters for the likelihood $p_\theta(x \mid z)$. The type of likelihood depends on the type of the initial data. For binary input data, a Bernoulli distribution is often used. For continuous data, the likelihood is a Gaussian distribution with a spherical covariance: $D_\theta(z) = (\mu_z, cI)$, where D_θ is a neural network, and $p_\theta(x \mid z) = f(x \mid \mu_z, cI)$, with I the identity matrix. The scalar c can be tuned to balance the VAE reconstruction and generation abilities according to Dai and Wipf (2019). If the decoder is well-trained, a point sampled from $p_\theta(x \mid z)$ should look like the input that constructed the posterior which z has been drawn. To allow for the back-propagation of the gradient despite the random draw of $z \sim \mathcal{N}(\mu_x, \Gamma_x)$, the *reparametrization trick* is used. A [id = r1]random vector ζ is sampled from a standard Gaussian $\zeta \sim \mathcal{N}(0, I)$, and z is formed with $z = \mu_x + \Gamma_x^{1/2} \times \zeta$ using the Cholesky decomposition. As a result, the use of a Gaussian posterior distribution is mandatory. The full architecture of the VAE is summarized on Fig. 3.

The VAE objective: Evidence lower bound (ELBO). The VAE framework relies on the *Variational Inference*, which seeks to find the best approximate $q_\phi(z \mid x)$ of $p_\theta(z \mid x)$. In other words, the VAE aims to minimize the Kullback–Leibler divergence $D_{KL}\left[q_\phi(z \mid x) \parallel p_\theta(z \mid x)\right] = \mathbb{E}_{z\sim q_\phi}\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right]$. One can show that this equality implies a lower bound on the desired log-likelihood $\log p_\theta(x)$:

$$\log p_\theta(x) \geq \mathbb{E}_{z\sim q_\phi}\left[\log p_\theta(x \mid z)\right] - D_{KL}\left[q_\phi(z \mid x) \parallel p(z)\right] = ELBO \tag{4}$$

Optimizing ELBO takes into account two phenomena:

- *maximizing the decoder log-likelihood* $\mathbb{E}_{z\sim q_\phi}\left[\log p_\theta(x \mid z)\right]$ over the observed data x, which ensures a reliable reconstruction.
- *minimizing the divergence between the latent distributions and the prior* $D_{KL}\left[q_\phi(z \mid x) \parallel p(z)\right]$ to be sure that the latent space distribution given by the encoder is as close as possible to the prior.

4.2. Model improvements

As it will be exposed in Section 5.2, the sole use of a VAE architecture is not sufficient to provide an efficient estimation method for the distribution of observed aircraft trajectories. Each structural component of the VAE has to be adapted to the generation of multivariate time-series with strong correlations from one data point to another. First, we present how information embedding through the dimensionality reduction has been improved with Temporal Convolutional Networks, and then how to adapt the generation process to these modifications with a Variational Mixture of posteriors.

4.2.1. Temporal convolutional networks

In the VAE architecture, dimensionality reduction is carried out thanks to the encoder and the decoder. The challenge is to find the best encoder/decoder functions that both reduce the dimension while keeping the loss of information minimal; i.e, being able to accurately reconstruct a sample from its embedding. In the context of aircraft trajectories, it is essential to be able to take into account the time dependency. Sequence modelling refers to the analysis of time series in Deep Learning. Suppose $X = (x_0, x_1, \dots, x_t, \dots, x_T) \in \mathbb{R}^{d_x \times T}$ a d_x-dimensional time-dependent input sequence of length T, such as the 3-dimensional position of an aircraft. Sequence modelling aims to capture the temporal dependencies within data in order to predict features $Y = (y_0, y_1, \dots, y_t, \dots, y_T) \in \mathbb{R}^{d_y \times T}$, where y_t only depends on previous observations $x_0, \dots, x_t$. The vector y_t is here the latent

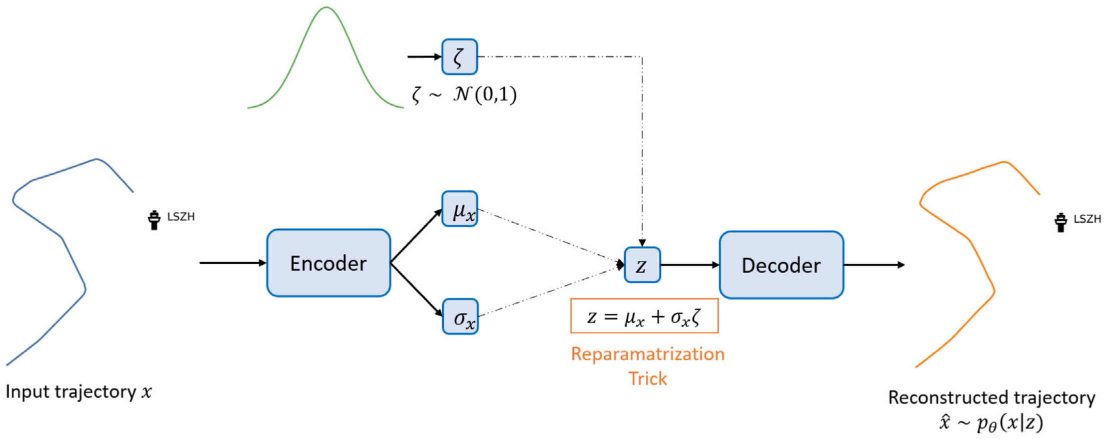

Fig. 3. Basic architecture of a Variational Autoencoder.

representation features that embody the temporal component of the sequence. While the Recurrent Neural Network (RNN) architecture is applied for sequence modelling in most cases (Goodfellow, Bengio, & Courville, 2016; Lazzara et al., 2022), Bai, Kolter, and Koltun (2018) suggest that CNN should also be considered a legitimate option due to many reasons: they are less complicated, less exposed to exploding or vanishing gradients, allow for parallel computation of outputs (unlike RNN), and can achieve cutting-edge performance. The authors exhibit a family of architectures called Temporal Convolutional Networks (TCN) that adapt general CNN for sequence modelling tasks.

TCNs are based on two fundamental principles: X and Y should have the same lengths T (but not necessarily the same number of channels d_x and d_y), and no information from the future can be used to predict the past. These principles are achieved through the use of *causal convolutional layers* with an adapted zero-padding. A convolutional layer is said to be *causal* if the output element of index t is convolved only from input elements of time t and earlier, as illustrated in Fig. 4(a).

The receptive field r of a causal convolutional layer describes both the memory of the layer and how far an output is affected by the past, and generally a full history coverage is desirable. Therefore, the output t should depend on all previous inputs $0, \dots, t$. *Dilatation* is then required to achieve a long effective history while keeping a reasonable amount of layers. The basic block architecture of the TCN displayed in Fig. 4(b) consists of stacking several causal convolutional layers with an increasing dilatation factor until the full history coverage is obtained. Eventually, Bai et al. (2018) combine the TCN block with the residual architecture developed by He, Zhang, Ren, and Sun (2016) to reduce the risk of exploding and vanishing gradients. This way, residual block outputs $o = \text{activation}(f(x) + x)$ instead of $o = \text{activation}(f(x))$, where f is the neural network. To be able to compare x and $f(x)$ which have the same lengths, but not necessarily the same number of channels, an optional 1×1 convolution is added to the TCN block to ensure the final element-wise addition receives terms of the same shape. The usage of a residual architecture does not change the approach conceptually, but shows significant performance gains for deep networks. The final TCN residual block is given by Fig. 4(c). Subsequently, the proposed VAE encoder and decoder are built with a superposition of TCN residual blocks.

TCN are a simple and efficient alternative to RNN to capture the temporal dependencies of timeseries. As shown in Section 5.1, TCN reduce by far the loss of information during dimensionality reduction and allow for a finer representation of trajectories in the latent space and a better reconstruction. However, the aggregated posterior distribution in the latent space becomes significantly more complex. It is therefore imperative to choose an adequate prior distribution in order to keep the generation abilities for new samples acceptable.

4.2.2. Variational mixtures of posteriors

It is common practice to use a standard Gaussian of dimension M as the prior distribution of the VAE as it makes the calculation of ELBO easier. However, such a simplistic prior might not be sufficient for the generation of satisfying results, since complexity was introduced in the latent space by using TCN encoders instead of Fully Connected Networks (FCN). Moreover, according to Hoffman and Johnson (2016), the training process is ruled by the following trade-off:

- *the reconstruction term of ELBO* forces the VAE to behave like a regular autoencoder (AE) by over-fitting points in the latent space. The variational posterior distributions $q_\phi(z \mid x)$ tend to have well-separated means and small variances to avoid overlaps from one distribution to another. Thus, it is easier for the decoder to reconstruct inputs.
- *the regularization term of ELBO* encourages overlaps between variational posteriors $q_\phi(z \mid x)$ by forcing them to approach the standard Gaussian prior. A regularization which is too weak forces the VAE to tend towards a simple AE, where all distributions of the latent space are practically point-like. Therefore, the generation with a continuous distribution is no longer relevant. In contrast, a regularization which is too strong collapses all the distributions to the same standard Gaussian prior, and thus no distinction can be made between trajectories in the latent space.

Consequently, the VAE must construct a latent space that is both diverse enough in the organization to allow for good reconstruction and also simple enough to be well covered by the prior. A Gaussian prior is not sophisticated enough to draw points in a latent space with a complex shape induced by the TCN encoder. The generated points are too far away from real embedded data, which leads to a rather poor generation behaviour. As suggested in Hoffman and Johnson (2016, p. 4), multimodal priors could "meet halfway" to satisfy both objectives of the trade-off. Subsequently, latent space can have a complex organization for reconstruction, while guaranteeing that the prior can still generate points in the right areas. It appears then that building a multimodal prior based on outputs from the encoder might allow for better coverage of the latent space while ensuring that points sampled within the prior are decodable. This idea is developed by Tomczak and Welling (2018) who present a new type of prior, called the *Variational Mixture of Posteriors* (VampPrior). It consists of setting the prior distribution as a Gaussian Mixture whose components are given by posterior distributions conditioned on learnable pseudo-inputs. In other words, an additional fully connected network learns the best inputs x_i^{pseudo} for $i = 1 \dots K$, where K is the number of pseudo-inputs to feed the encoder. Their corresponding posterior distributions in the latent space $q_\phi(z \mid x_i^{pseudo})$, $i = 1 \dots K$, are used as components for the prior. K should be chosen so that it is large enough compared

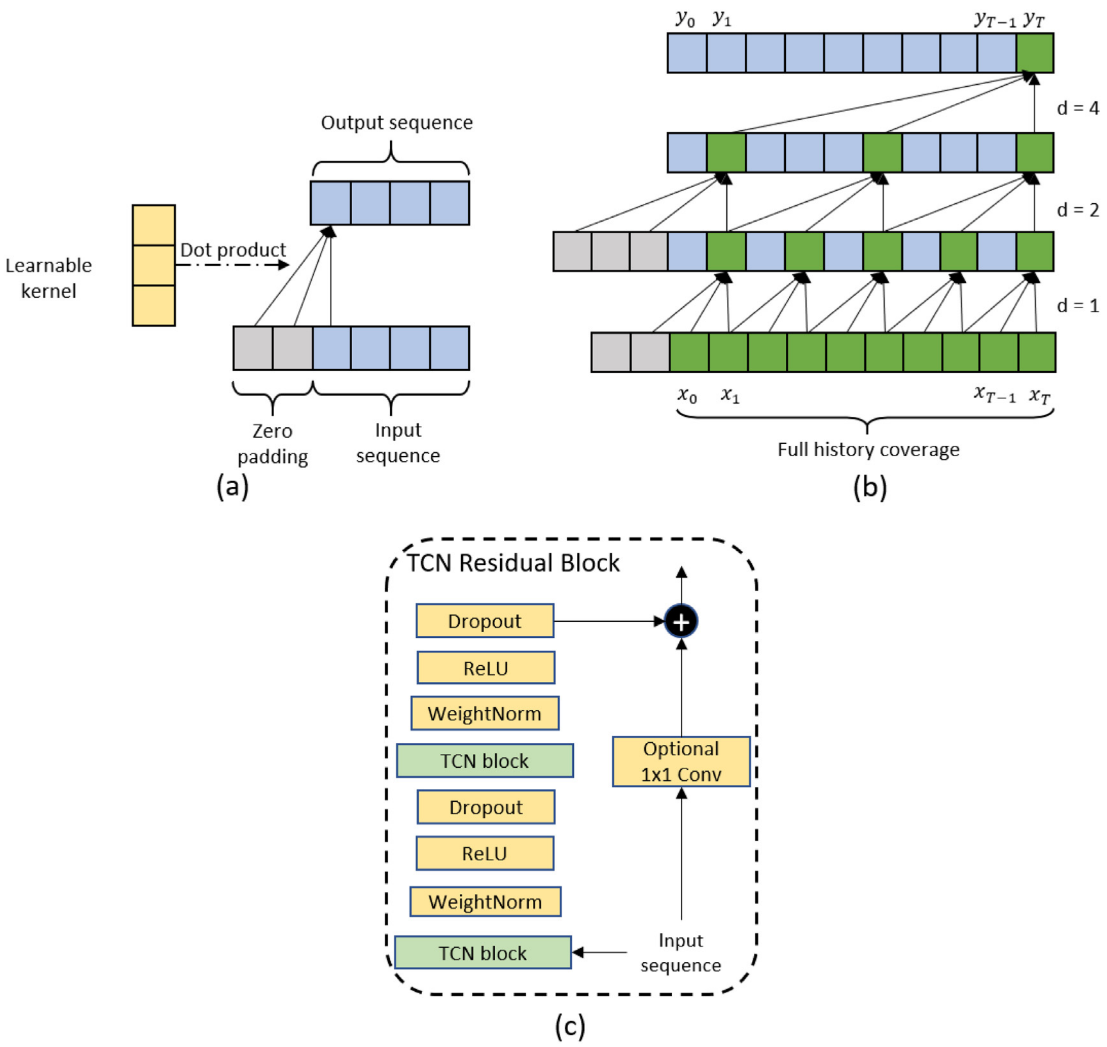

Fig. 4. Architectural elements of a TCN. (a) represents a 1-dimensional convolutional layer with a kernel size of 3. (b) represents 4 1-dimensional convolutional layers with a kernel size of 3 and a dilatation factor of 2, called a TCN block. (c) represents the full structure of a TCN residual block, which is the base architectural element of a TCN.

to the size of the training dataset to cover the latent space efficiently, but also reasonably small to avoid a long learning time. Then, the prior probability distribution function is:

$$p(z) = \frac{1}{K}\sum_{i=1}^{K} q_\phi(z \mid x_i^{pseudo}) = \frac{1}{K}\sum_{i=1}^{K} f(z \mid \mu_{VP,i}, \sigma^2_{VP,i}) \quad (5)$$

where $\mu_{VP,i}, \sigma^2_{VP,i}$ are the parameters of the ith Gaussian component in VampPrior. The prior distribution is no longer specified before but learned during the training of the VAE through the pseudo-inputs generator, which is updated in such a way that the encoder will output the best component parameters to minimize the regularization term in ELBO.

4.3. VampPrior temporal convolutional variational autoencoder for trajectory generation

The detailed architecture of the proposed temporal convolutional variational autoencoder (TCVAE) is shown in Fig. 5, and can be found on GitHub.[4] The latent space dimension M, which is the size of each vector $\mu_{post,i}, \sigma_{post,i}$ (corresponding to one posterior) and $\mu_{VP,i}, \sigma_{VP,i}$ (corresponding to one VampPrior component) has been set to $M = 64$. The number of pseudo-inputs is set to $K = 1000$. Despite M and K are hand-tuned parameters, those values might work for most airport configurations as approach procedures at Zurich airport are rather complex. If the training trajectories have simpler shapes, one might want to reduce training time by setting a lower value of M as a latent space of smaller dimension could be sufficient. K corresponds more to the number of different procedures. For airports with less diverse trajectories, one might want to set a lower value for K. The model has been trained on 1000 epochs, using an Adam optimizer with a learning rate of 0.001 that is being halved every 200 epochs. The batch size has been set to 500. Data are normalized with a MinMax scaler to get values between −1 and 1. The training on a computer without GPU takes about 14 h. However, once the model is trained, the generation is instantaneous.

In addition to improving VAE learning, VampPrior also allows controlling the type of samples to be generated. Once decoded, the pseudo-inputs represent synthetic trajectories whose shape is representative of the region in which they are located in the latent space. Thus, by drawing new points around a particular pseudo-input, we obtain resembling trajectories:

- *the mean*, $\mu_{VP,i}$, defines the region of the latent space in which the selected ith VampPrior component simulates points. Once decoded, the mean represents a synthetic trajectory having the shape of observed trajectories located in the same area of the latent space.

[4] https://github.com/kruuZHAW/deep-traffic-generation-paper.

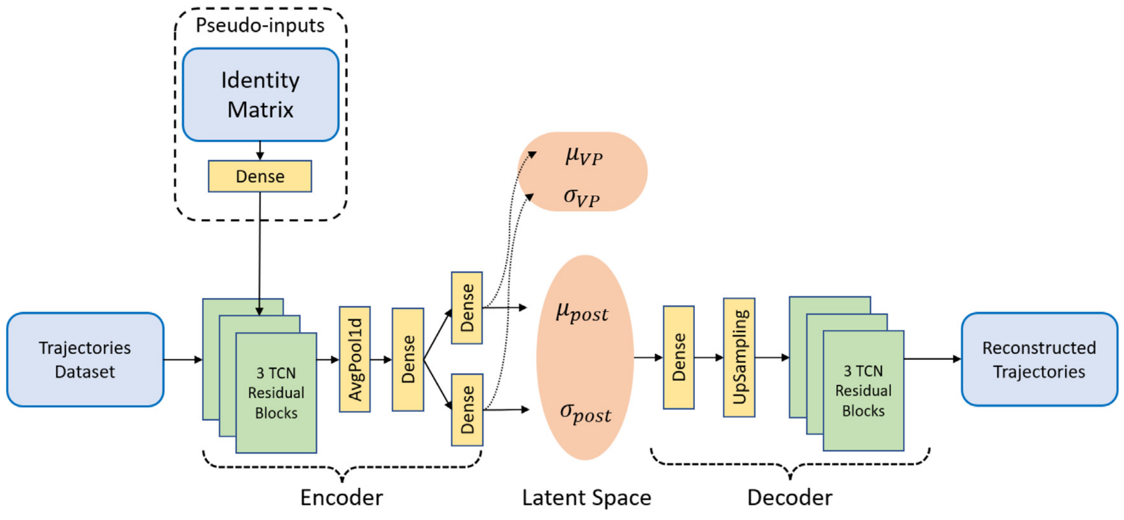

Fig. 5. Structure of the proposed VampPrior TCVAE. μ_{VP}, σ_{VP} stand for the VamPrior parameters, and $\mu_{post}, \sigma_{post}$ for the ones of the posteriors.

- *the variance*, $\sigma^2_{VP,i}$ (which is also diagonal here), defines how far from $\mu_{VP\,i}$ the associated Gaussian component draws points. A small variance leads to the simulation of points close to the mean in the latent space, and their associated decoded trajectories will be very similar to the decoding of $\mu_{VP\,i}$. Alternatively, a bigger variance leads to a greater diversity of the synthetic trajectories around the mean, sometimes at the cost of degradation in the realism of the generated synthetic trajectories.

Therefore, the generation process of synthetic trajectories is done in three steps: (i) the identification of the region of the latent space where the trajectories have the characteristics we want to generate, (ii) the search in the selected region for a pseudo-input, (iii) the random draw of latent points in the corresponding component of VampPrior, $\mathcal{N}\left(\mu_{VP,i}, \sigma^2_{VP,i}\right)$. Once decoded, they provide random synthetic trajectories around the selected pseudo-input.

5. Results and discussion

5.1. VAE architecture improvements

This section highlights how the architecture modifications of the VAE have improved its learning phase for trajectory modelling task. First, results show that the TCVAE presents improved reconstruction abilities compared to a FCVAE (Fully Connected Variational Autoencoder). Although not the main focus of the study, reconstruction is an important component to consider. The reconstruction gives insight about how well the information is encoded from the initial trajectory to its latent space representation. If the reconstruction is poor, the VAE is not able to capture the characteristics of aircraft trajectories because too much information is lost during the reduction of dimension. We also analyse the organizational differences in the respective latent spaces. Then, the examination of ELBO show the significant improvements provided by VampPrior TCVAE over basic VAE architectures.

5.1.1. Contribution of temporal convolutional networks

It is important to note that FCVAE and TCVAE are difficult to compare. As such, the two architectures are comparable neither in their number of parameters nor in their learning time. The comparison presented hereafter is based on getting the best possible optimization of ELBO in a reasonable learning time. The encoder of the FCVAE is composed of an FCN of three layers (218, 128, 64) and the TCVAE encoder has 4 residual blocks with 64 output channels each, which ensures that full history coverage is met. Both have a latent space dimension of 64, trained on 1000 epochs, and their decoders have the same architectures as the encoders but are reversed. It is obvious that TCVAE has a much better reconstruction performance, and can handle not only simple straight trajectories but also more complex ones than FCVAE, as illustrated with an example in Fig. 6 (which will be confirmed by the analysis of ELBO in Section 5.1.2).

The reconstruction abilities differences between TCVAE and FCVAE are also highlighted in Figs. 7 and 8. They represent the latent space clustering associated with the corresponding reconstructed trajectories for FCVAE and TCVAE respectively. It is obvious that FCVAE in Fig. 7 strongly struggles to reconstruct accurately the observed flows of trajectories compared to the TCVAE 8. Moreover, TCVAE seems to take into account more refined characteristics of trajectories whereas FCVAE groups trajectories mostly on direction of arrival. For instance, purple points for TCVAE in Fig. 8 represent direct approaches from the east, whereas green ones correspond to complex approaches from the same direction. Moreover, the TCVAE latent space contains more clusters than the one of FCVAE. It suggests that the projection axes provided by the TCVAE are much more efficient to render non-obvious intrinsic characteristics of trajectories, and contain way more information to enable an accurate reconstruction.

5.1.2. Learning improvements of VampPrior TCVAE compared to basic VAE architectures

Table 1 compares the training loss ELBO defined in Eq. (4) for different VAE architectures. To this end, the resulting reconstruction term (which should be maximized) and the KL-divergence (which should be minimized) are listed in the table. FCVAE is much less efficient than TCVAE in terms of reconstruction, but shows a good match between the aggregated posterior and the Gaussian prior. This is because FCNs might not be able to render the complicated characteristics of trajectories in their latent representations. Thus, the organization of the latent space is more elementary, and a simple Gaussian prior distribution is sufficient to have good coverage of the aggregated posterior distribution. However, the loss of information during dimensionality reduction is significant, and trajectories cannot be reconstructed properly. TCVAE significantly improves the reconstruction ability, but also provides a more complex latent space organization. Consequently, a simple Gaussian prior is unable to cover the latent space properly anymore, and points generated with the prior are not located in the same areas as points from the aggregated posterior. Thus, they cannot be handled by the decoder. This results in the generation of syn-

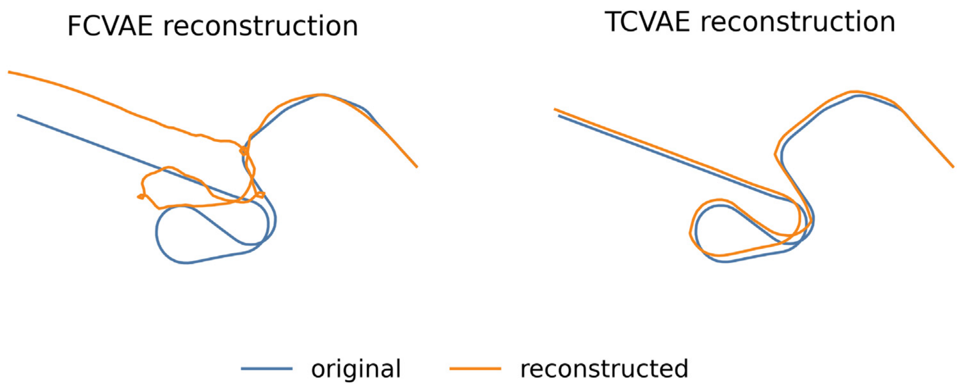

Fig. 6. Reconstruction of FCVAE and TCVAE for a complex trajectory with a holding pattern.

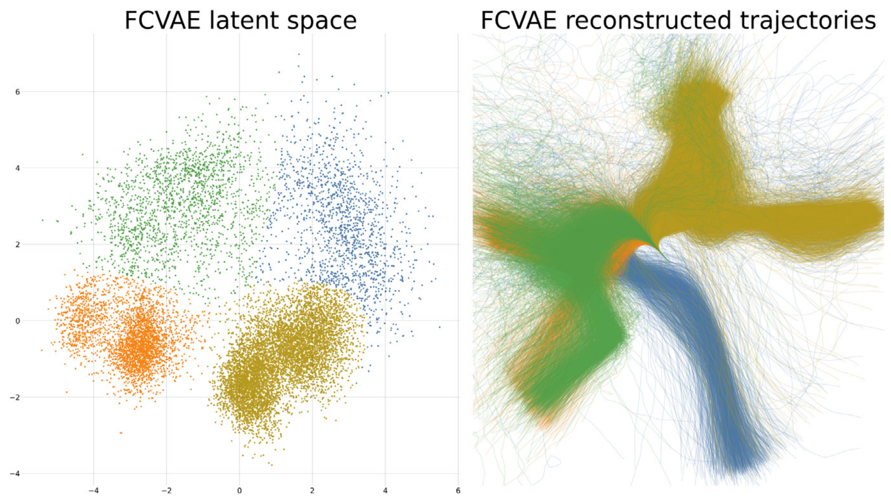

Fig. 7. Gaussian clustering made in the PCA projection of latent space of observed trajectories for the FCVAE.

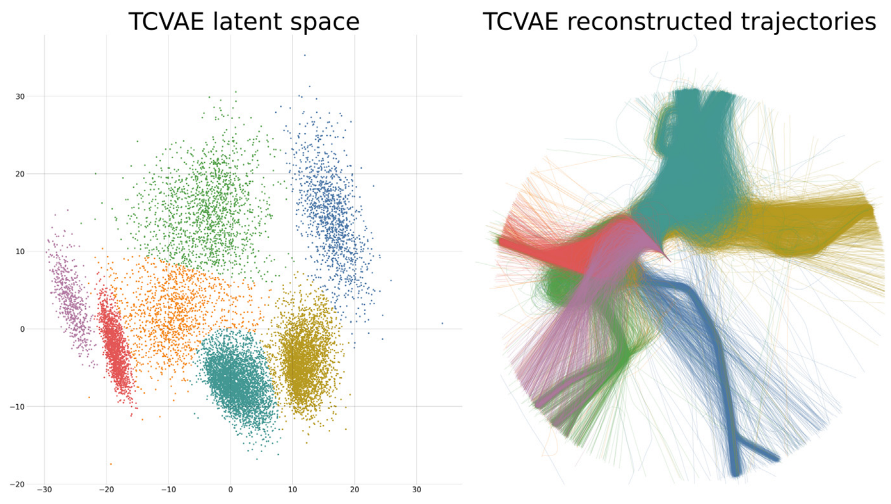

Fig. 8. Gaussian clustering made in the PCA projection of the latent space of observed trajectories for the TCVAE.

thetic trajectories which are not realistic. Using VampPrior overcomes this difficulty; it keeps the reconstruction power of the TCVAE while improving the match between the latent space distribution and the prior. Consequently, points generated in the latent space are closer to the aggregated posterior and can be decoded efficiently through the decoder to give realistic trajectories.

Table 1 clearly states that the use of a basic VAE architecture is not sufficient to solve this generation problem. Moreover, only improving the reconstruction ability with TCN encoder and decoder tends to significantly impair VAE learning by increasing the KL-divergence. To counter this phenomenon, a more complex prior distribution such as VampPrior must be used.

Table 1
ELBO metrics comparison for VAE architectures after 1000 epochs.

VAE architecture	Reconstruction term	KL-divergence	ELBO
Fully Connected encoder, Gaussian Prior	963	32	931
Temporal Convolutional encoder, Gaussian Prior	2413	217	2196
Temporal Convolutional encoder, VampPrior	2438	169	**2269**

Table 2
e-distance between observed and synthetic trajectories.

Estimation method	E-distance
Gaussian Mixture 24 components	0.0590
Fully connected encoder, Gaussian prior, VAE	0.9265
Temporal convolutional encoder, VampPrior VAE	0.0103

5.2. Evaluation of the generated trajectory quality

In Section 5.1, we demonstrated that the proposed VampPrior TCVAE architecture significantly improves the learning performances compared to a basic FCVAE architecture for the modelling of 4-dimensional aircraft trajectories. We now analyse the quality of the generated trajectories. The VampPrior TCVAE is compared with other generative models, such as Gaussian Mixtures mentioned in Murça and de Oliveira (2020), on its ability to estimate the distribution of observed trajectories and to produce realistic synthetic trajectories. We propose here to analyse the quality of the generated synthetic trajectories regarding the two following criteria: the preservation of the statistical properties of the observed trajectory dataset and the flyability of the generated trajectories. For this purpose, we compute the following metric for all trajectories, considering both criteria and all four trajectory generation methods.

Table 2 summarizes the *e-distance* between observed and synthetic trajectory flows. Introduced in Székely, Rizzo, et al. (2004), the *e-distance* provides a measure of the distance between the respective distributions of two sets of random vectors. It tends towards a positive constant if the two samples are identically distributed, and tends to infinity otherwise. Therefore, *e-distance* can be used here to determine which generation method produces the synthetic trajectories whose distribution is closest to the observed one. It has been applied to two sets of observed and synthetic trajectories. These sets each consist of 3000 trajectories that are described with vectors of 800 features. The results have been averaged over 100 retrials for each generation method.

The Gaussian mixture model (GMM) presented in Murça and de Oliveira (2020) represents the baseline against which VAEs are compared. The poor results shown by FCVAE are the direct consequences of the shortcomings outlined in the analysis of Table 1. Despite a good match between the aggregated posterior and the prior, it does not have the necessary reconstruction power to produce realistic trajectories, and thus to estimate accurately the target distribution. The use of a VAE is only of significant interest if it has been specifically adapted for the trajectory generation problem. VampPrior TCVAE overcomes this difficulty, while guaranteeing satisfying generation performances. First, its architecture can reconstruct trajectories very well, but its prior is also complex enough to guarantee that synthetic points in the latent space are similar to true points. Moreover, the VAE is based on a significantly larger number of parameters to describe the distribution to be estimated. Thus, it provides a much more complex and refined parametric model than the one given by GMM. As a result, VampPrior VAE is by far the most accurate method presented here to estimate the statistical distribution of observed trajectories, by improving by a factor of 6 the results obtained for the GMM. VAE constitutes real improvements for the generation of multivariate trajectories only when all model assumptions have been reviewed and adapted to the problems encountered.

We have shown that the VampPrior TCVAE is the method that produces synthetic trajectories whose statistical distribution is closest to the observed trajectories. Even though the goodness-of-fit is satisfying, we also have to ensure that the generated synthetic trajectories can be flown by an actual aircraft. In this context, Olive et al. (2021) present a framework to assess the quality of synthetic trajectories by replaying them in a simulator, which takes flight mechanics equations into account. We replayed our generated trajectories in the open-source air traffic simulator *BlueSky* (Hoekstra & Ellerbroek, 2016). A given synthetic trajectory is considered physically realistic if the distance between a trajectory and its replayed counterpart is small. We compute nine trajectory distances implemented in the `traj-dist` Python library.[5] Fig. 9 shows the cumulative distribution of the Dynamic Time Wrapping (DTW) (Besse, Guillouet, Loubes, & François, 2015) and the Symmetric Segment-Path Distance (SSPD) (Berndt & Clifford, 1994) between synthetic trajectories, and their simulator-generated version. More precisely, trajectories were divided into segments in which the aircraft are aligned with existing navigational beacons as well as flying at a constant speed and altitude. Then, BlueSky generates the corresponding ATC instructions to recompute each trajectory. This metric heavily relies on the assumption that BlueSky is efficient enough to exactly recompute observed trajectories. That is not necessarily the case, as Bluesky is not able to handle runway alignment or self-intersecting trajectories. To compensate for this deficiency, a reference metric is also calculated on observed samples. A generation method is then considered realistic if its scores are close to those obtained on the reference.

Fig. 9 illustrates that the VampPrior TCVAE is the generation method that produces the most realistic trajectories, whereas the basic FCVAE leads to the weakest results. This is consistent with the observations made on the e-distance. Indeed, the method that produces the most statistically distant trajectories produces also the least realistic ones. Moreover, as simulated trajectories have been recomputed based on ATC instructions, Fig. 9 also shows that the VampPrior TCVAE generates the most relevant trajectories from an operating point-of-view. The GMM gives results slightly better than the FCVAE, but significantly weaker than the TCVAE. In their analysis, Murça and de Oliveira (2020) the generated trajectories were evaluated by experts (ATC, pilots, etc.), and most of them could not tell the difference between a true trajectory and a synthetic one. Mathematically speaking, our VampPrior TCVAE produces trajectories that are even closer to reality.

5.3. Air traffic modelling with VampPrior TCVAE

5.3.1. VampPrior within the latent space of the VampPrior TCVAE

Section 5.2 shows that the VampPrior TCVAE outperforms other density estimation methods in terms of goodness-of-fit and quality of the generated samples. Subsequently, this section provides an analysis of how trajectories are encoded in the VampPrior TCVAE latent space. Fig. 8 depicts that clustering of the latent space organization is possible to group trajectories by upcoming directions and shape. Thus, one can select the type of trajectories to generate by focusing on a specific cluster in the latent space. For instance, a point sampled in the green region of Fig. 8 will correspond to a synthetic trajectory approaching runway 14 of Zurich Airport from the west with a non-direct approach, whereas the purple region describes trajectories coming from the west with a rather straight-in approach.

[5] https://github.com/bguillouet/traj-dist.

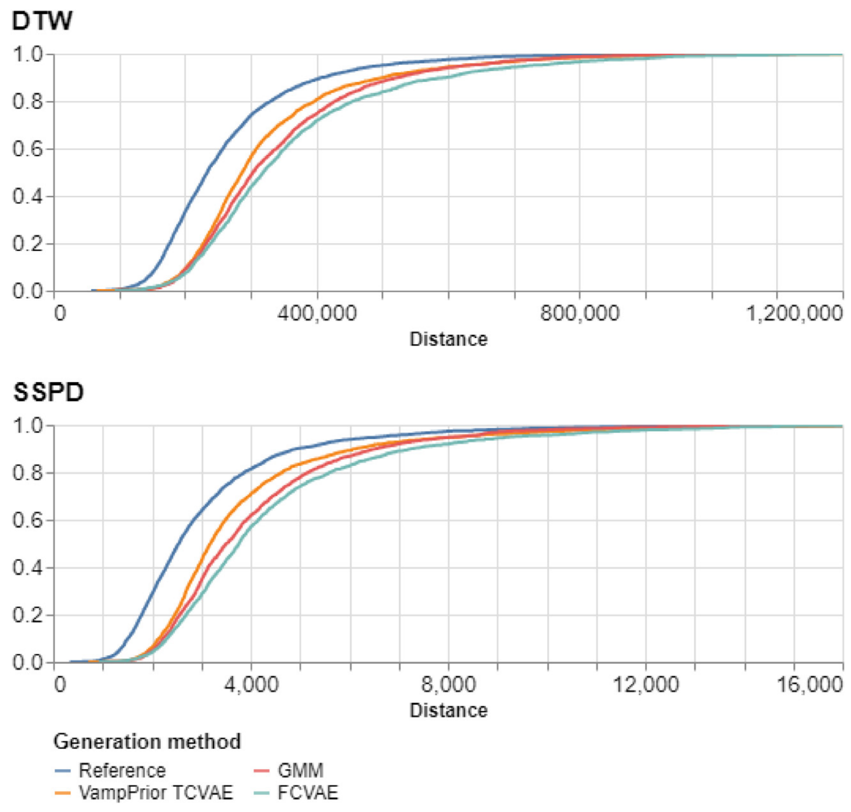

Fig. 9. Cumulative distribution for DTW and SSPD between synthetic trajectories and their simulated versions in Bluesky.

To ensure a realistic generation behaviour, the points drawn in the prior must be located close to the points drawn in the aggregated posterior, representing the observed trajectories. In Fig. 10, generated points from the prior seem to be distributed identically to points from the posterior distribution. In Fig. 11, we detail how the VampPrior components cover the latent space. The left part of Fig. 11 displays the location of each mean $\mu_{VP,i}$, $i = 1 \dots K$ of the VampPrior components (equivalently called pseudo-inputs) in the latent space. The colour of the points indicates the variance of the components. A yellow point represents a VampPrior component with high variance, whereas a dark blue point represents a small variance. The right part of Fig. 11 depicts the decoding of the VampPrior means. The VampPrior covers well the whole latent space and respects the densities of the regions, as there are more pseudo-inputs in denser areas. Furthermore, it seems that the pseudo-inputs can be classified according to their variance. The ones in denser areas, such as the orange pseudo-input, have lower variances and are mostly associated with simple types of trajectories, just as suggested by the analysis of the latent space in Fig. 8. As a result, points sampled in these components are close to the mean, and then should look very similar to the decoded pseudo-input. On the contrary, in sparser regions, the variance is usually larger to cover a wider area. Points can be sampled far away from the mean, and thus can lead to the generation of trajectories significantly different from the pseudo-input.

5.3.2. *Generation with VampPrior TCVAE*

Fig. 12 displays the generation of synthetic points in two specific VampPrior components. The orange flow corresponds to a component located in a dense area, whereas the blue flow corresponds to a component in a sparser area. The former is associated with an arrival trajectory which is flown in a rather straightforward way, i.e. without holding patterns and with very limited influence of air traffic control. In this region of the latent space, the VampPrior component does not cover any empty area. Consequently, generated synthetic trajectories are realistic, even though a wide diversity is observed. It is conceivable that this generated flow corresponds to a type of approach, where the pseudo-input describes the ideal path, and other trajectories contain high levels of uncertainties such as aircraft type or weather. The blue flow is associated with an arrival trajectory which is flown both with a holding pattern and with a substantial influence of air traffic control. The considered region of the latent space is less dense, and the VampPrior component covers some empty areas. Generated points can be isolated, and the closest observed points can correspond to very different types of trajectories. As a result, these kinds of VampPrior components generally generate complex trajectories with self-intersecting points, that can be very disparate. This great diversity may lead to certain trajectories being less realistic from an operational point of view. Loops or turns can be located at different locations with varying sizes and curvatures, which means that certain of these configurations are very unlikely to be observable in reality, but still interesting to generate.

Trajectories with complex shapes are relatively rare and different from most other trajectories. As the dimension of the latent space is large, points located in sparse regions are very distant. Thus, the VampPrior components that cover these locations must have high variances, at the risk of producing points in deserted parts of the latent space where no posterior distribution is close. Consequently, once decoded, these points give unusual trajectories, which often look physically realistic, but can be far from the actual aircraft operations observed in reality. Indeed, as the region is sparse, a generated point can be found between very different observed trajectories. Subsequently, as the latent space is continuous, it inherits the characteristics of its nearest neighbours, even if the result is operationally very unlikely. Reducing the size of the latent space can decrease the effect of the spacing between points. However, it will also reduce the ability of the decoder to reconstruct trajectories well.

Nevertheless, both blue and orange generations presented in Fig. 12 seem to have realistic altitude and ground speed profiles, as displayed in Fig. 13. The VampPrior TCVAE can take into account non-trivial behaviours, such as non-monotonic ground speed and altitude profiles,

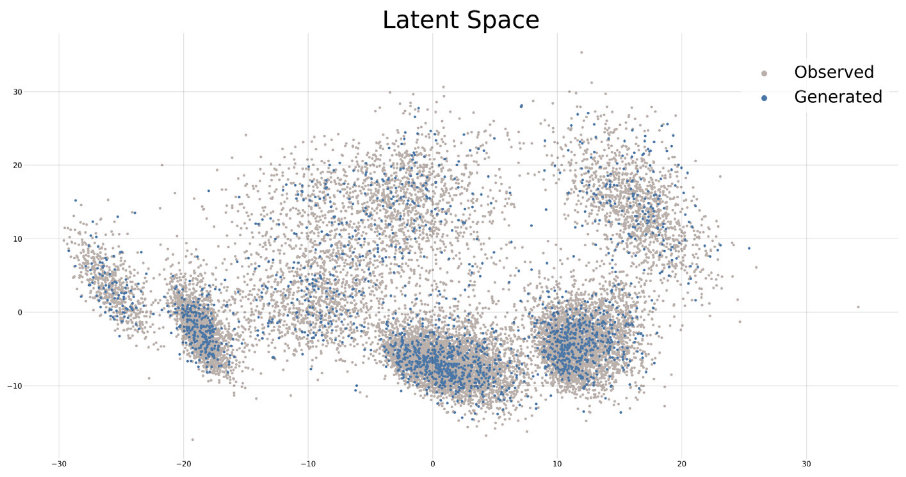

Fig. 10. Point sampled in the aggregated posterior (grey) and in the prior (blue) in the latent space.

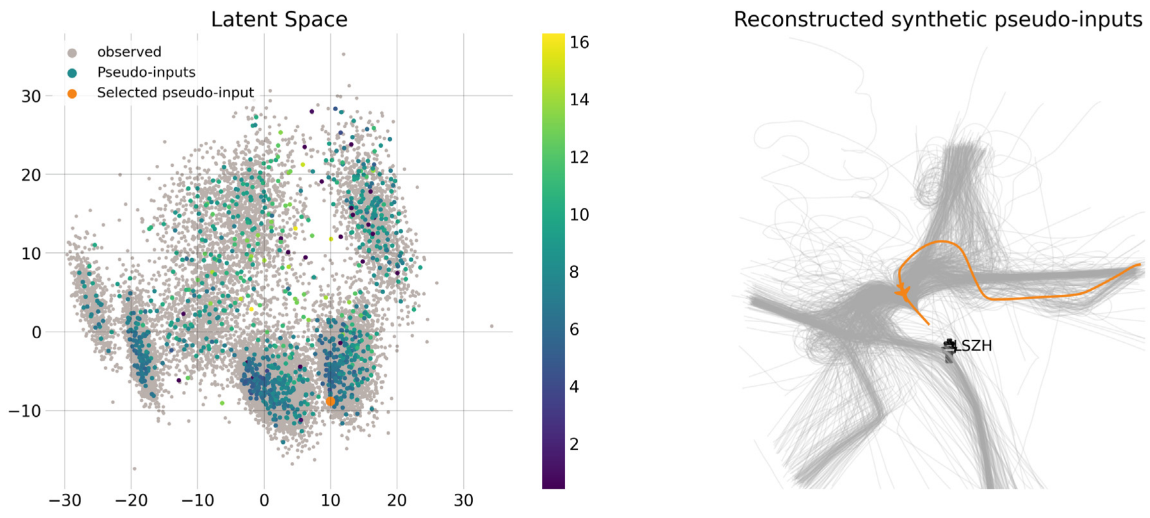

Fig. 11. Left: Pseudo-inputs representing the Gaussian mixture components of VampPrior. The location of the point represents the mean, and the colour the variance. Right: Decoding of the VampPrior pseudo-inputs.

which is one of the limitations of the GAN suggested by Jarry et al. (2019). The model even learns by itself that there is a speed limit of 250 kts under 10,000 ft. Besides that, ground speeds for the orange generation are even divided into two different types of approaches at the beginning of the profile, which might be associated with different aircraft types. Finally, the TCVAE is also capable of generating trajectories with different durations and lengths, making it very suitable to render observed approaches in various environmental conditions. In summary, Figs. 12 and 13 show that the VampPrior TCVAE can generate a wide diversity of physically realistic trajectories that can have complex behaviour patterns. However, this great diversity might be sometimes associated with a lack of operational realism, for example by generating holdings in unlikely places.

The quality of the generated synthetic trajectories heavily relies on the quality of the considered pseudo-input. As such, pseudo-inputs can be classified into three groups. The first one, observed on Fig. 12 is composed of pseudo-inputs located in the denser areas of the latent space. They are by far the most numerous and are surrounded by observed trajectories with simple patterns. In these regions, the posterior distributions are largely overlapping, which results in a compact latent space. VampPrior components can cover the area more easily, and points drawn within these components are very similar to points sampled in the aggregated posterior. The corresponding decoded synthetic trajectories are then very realistic from a physical and operational point of view, while still maintaining a fair amount of variability. The second group is composed of pseudo-inputs located in sparser areas of the latent space, where complex trajectories with loops and holding patterns are observed. An example is given with the orange flow on Fig. 14. The size of this group is way smaller than the first one, but it still represents a significant part of the pseudo-inputs. In the dataset of observed trajectories, self-intersecting trajectories are rare and can take many forms. Consequently, the corresponding regions of the latent space are way sparser. Additionally, the distance from one point to another can be substantially larger (especially in high dimensions because distances are dilated). Corresponding VampPrior components have more difficulties to cover properly these areas. Moreover, they might generate points rather far away from the aggregated posterior. As a result, pseudo-inputs have the characteristics of surrounding points but might present defects. Some corresponding synthetic trajectories might be unlikely from an operational point of view, even though most of them stay physically relevant. However, this is not necessarily a drawback, especially when we study the influence of an unusual trajectory on how traffic is organized in the airspace. Finally, the third group is composed of non-realistic pseudo inputs. And example is the blue flow on Fig. 14. It is located in a very sparse area, and the corresponding VampPrior component has a very large variance. They are the consequences of the optimization of the VAE objective, since they are created to have the best match between the aggregated posterior and the prior according to the KL-divergence. Even though the

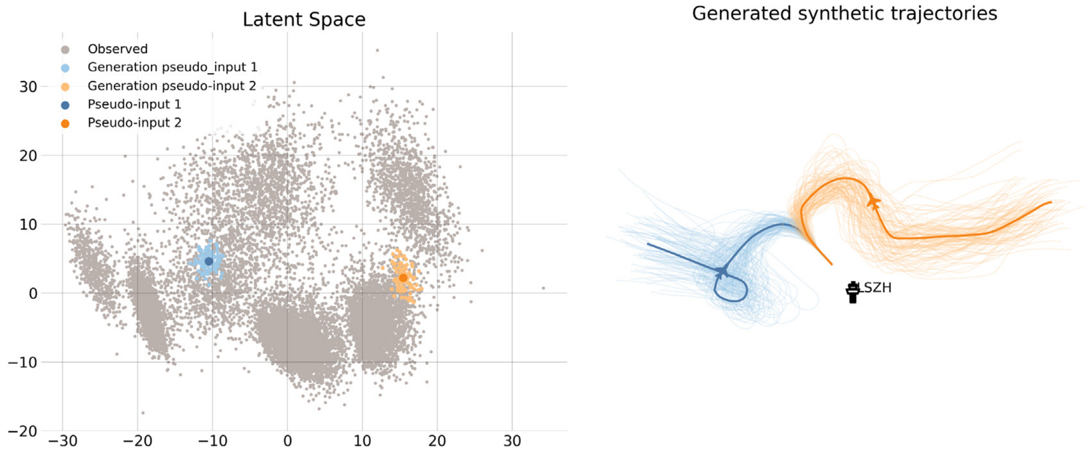

Fig. 12. Generation of synthetic trajectories within two pseudo-input components.

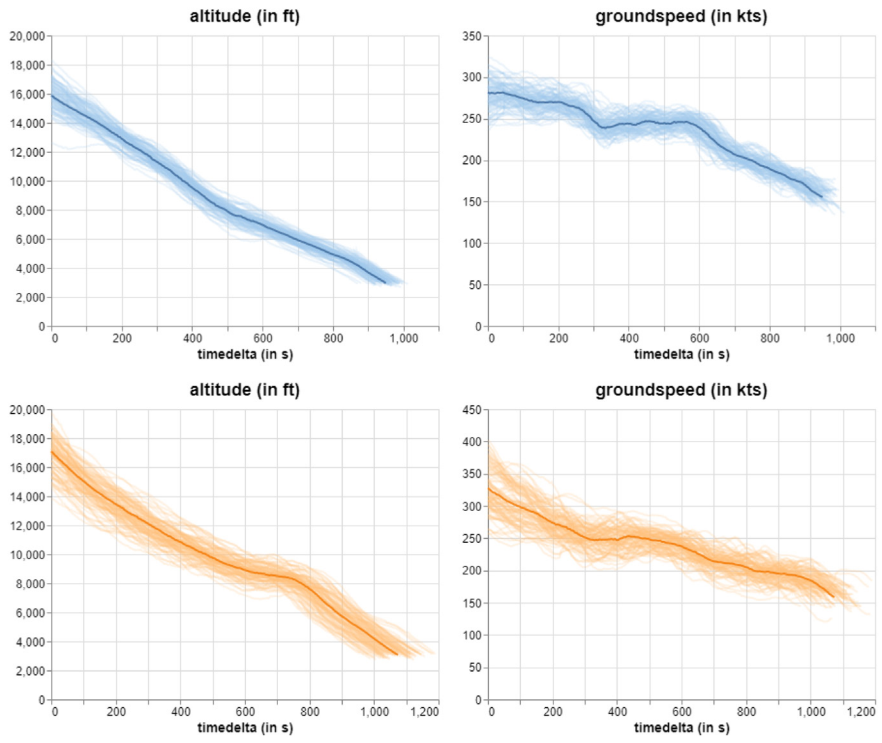

Fig. 13. Altitude and ground speed profiles of the generated trajectories from pseudo-input 1 (blue) and pseudo-input 2 (orange).

third group of pseudo-inputs is mathematically relevant, it generates non-realistic trajectories both from a physical and from an operational perspective, because of the missing constraints on their shapes.

6. Conclusion and outlook

We deconstructed the structural elements of the VAE to come up with a novel data-driven approach to generate synthetic trajectories. Such generative model is of great interest as it allows creating an arbitrarily large trajectory dataset from procedures where only a limited amount of observations is available. The TCVAE architecture is powerful enough to synthesize the temporal information from trajectories with complex characteristics in a latent space of smaller dimension. Coupled with VampPrior that enhances the generation abilities of the VAE by providing a more accurate prior distribution, the model can mimic the distribution of observed 4-dimensional trajectories to generate synthetic ones. The model fulfils all expectations mentioned in the introduction:

- Real and synthetic trajectories share the same statistical distribution, as suggested by the e-distance evaluation.
- Synthetic trajectories look realistic from a physical point of view according to the metric developed in Olive et al. (2021).
- We show that the generation process can provide a wide diversity of trajectories, including some that might never be observed from an operational point of view. This reflects the ability of the VAE to account for the uncertainty present in trajectories due to weather, aircraft performance, ATC actions or human factors. The means of each component of VampPrior represent the nominal behaviour of the route, whereas covariance matrices represent deviations due to uncertainty. The greater the covariance of a pseudo-input is, the more uncertainty we can expect from this router, and thus, the more diversified are the generated trajectories. On the contrary, a pseudo-input with a small covariance correspond to routes with few uncertainty. Although uncertainty is well rendered, we cannot identify its origin. It is possible to generate trajectories with unusual behaviour, but impossible to attribute it to a specific

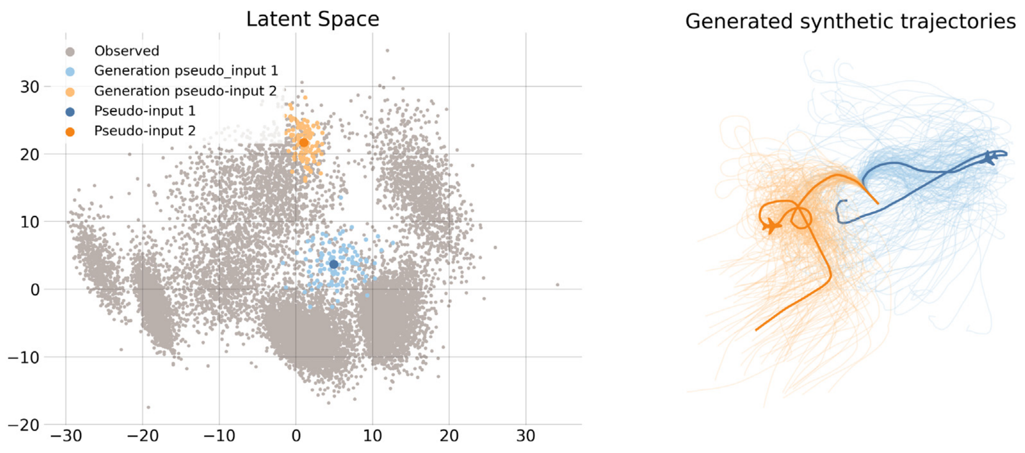

Fig. 14. Generation of synthetic trajectories from low quality pseudo-inputs.

source such as weather or ATC. An important limitation of our model is that we cannot generate two trajectories that we know are subject to the same sources of uncertainty.

- By selecting the appropriate VampPrior component of the Gaussian mixture prior, it is possible to control the type of synthetic trajectories to be generated, i.e. their direction, shape, etc., and to filter unrealistic generated trajectories.

Unlike most previous methods that focus on simple one or two-dimensional patterns (Eckstein, 2010; Henry et al., 2013; Jacquemart & Morio, 2013), the proposed model is capable of providing fully described trajectories in 4 dimensions. Moreover, it outperforms the GMM estimation from Murça and de Oliveira (2020). The model has been trained on a particularly complex set of approaching procedures at Zurich airport, and it makes no doubt that the VampPrior TCVAE will perform equally well on other airports by changing the training dataset. It can also be directly applied to other types of trajectories such as departures, en-route or even missed-approaches. However, missed-approaches studies might require a special treatment as they are rarely observed. Krauth et al. (2021) treated the problem by specifically constructing a dataset of go-around trajectories. One might also want to use the dataset provided by Monstein, Figuet, Krauth, Waltert, and Dettling (2022). A limitation of our method is that it is probably not able to randomly generate go-arounds among regular approaches. Finally, the distribution of observed trajectories is explicitly described in the latent space by the aggregated posterior. As a result, it is possible to evaluate the likelihood of a newly generated point. This information can be useful when using Monte Carlo methods for collision risk estimation such as *Importance Sampling* (Tokdar & Kass, 2010) or *Subset Simulation* (Au & Beck, 2001).

As mentioned in Section 4, the Mean Field Assumption is more than just a simplification to speed up the learning phase. It makes the dimensions of the latent space orthogonal and allows the empirical deduction of the role of each of the dimensions by modifying them independently. For example, by varying only the first component of the latent representation of a trajectory, it is possible to change the direction of arrival. This can be viewed as a very primary form of disentanglement, which focuses on controlling the decomposition of trajectories in the latent space. However, the encoder acts as a black box and one cannot choose in advance the linkage between the characteristics of a trajectory and the digits of its latent representation. To perform proper disentanglement, Chen, Li, Grosse, and Duvenaud (2018) change the functional form of ELBO to force the algorithm to focus on the Total-Correlation term that influences how dimensions are correlated in the latent space. Karaletsos, Belongie, and Rätsch (2015) introduce a similarity metric between inputs to identify what the essential latent dimensions should be. Disentanglement learning is nowadays a rather popular topic, and using it in future works could allow us to have even stronger control over the type of synthetic trajectories one wishes to generate.

CRediT authorship contribution statement

Timothé Krauth: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Adrien Lafage:** Methodology, Software, Investigation, Resources, Data curation, Writing – review & editing. **Jérôme Morio:** Conceptualization, Methodology, Formal analysis, Resources, Writing – review & editing, Supervision. **Xavier Olive:** Conceptualization, Methodology, Formal analysis, Investigation, Resources, Writing – review & editing, Visualization, Supervision. **Manuel Waltert:** Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work was supported by the Swiss Federal Office of Civil Aviation, grant number BAZL SFLV 2018-037. All authors approved the version of the manuscript to be published.

References

Asperti, A., & Trentin, M. (2020). Balancing reconstruction error and Kullback-Leibler divergence in Variational Autoencoders. *IEEE Access*, *8*.
Au, S.-K., & Beck, J. L. (2001). Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic Engineering Mechanics*, *16*(4), 263–277.
Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271.
Berndt, D. J., & Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD Workshop, Vol. 10* (pp. 359–370). (16).
Besse, P., Guillouet, B., Loubes, J.-M., & François, R. (2015). Review and perspective for distance based trajectory clustering. arXiv preprint arXiv:1508.04904.
Chen, R. T., Li, X., Grosse, R. B., & Duvenaud, D. K. (2018). Isolating sources of disentanglement in variational autoencoders. *Advances in Neural Information Processing Systems*, *31*.
Dai, B., & Wipf, D. (2019). Diagnosing and enhancing VAE models. arXiv preprint arXiv:1903.05789.
Delahaye, D., Puechmorel, S., Tsiotras, P., & Féron, E. (2014). Mathematical models for aircraft trajectory design: A survey. In *Air traffic management and systems* (pp. 205–247). Springer.

Eckstein, A. (2010). Data driven modeling for the simulation of converging runway operations. In *Proceedings of the 4th international conference on research in air transportation*.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, *27*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Henry, M., Schmitz, S., Revenko, N., & Kelbaugh, K. (2013). A Monte Carlo simulation for evaluating airborne collision risk in intersecting runways. In *Proceedings of the AIAA modeling and simulation technologies (MST) conference* (p. 4598).

Hoekstra, J. M., & Ellerbroek, J. (2016). Bluesky ATC simulator project: an open data and open source approach. In *Proceedings of the 7th international conference on research in air transportation*.

Hoffman, M. D., & Johnson, M. J. (2016). ELBO surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in advances in approximate bayesian inference, NIPS, Vol. 1*. (2).

Jacquemart, D., & Morio, J. (2013). Conflict probability estimation between aircraft with dynamic importance splitting. *Safety Science*, *51*(1), 94–100.

Jacquemart, D., & Morio, J. (2016). Adaptive interacting particle system algorithm for aircraft conflict probability estimation. *Aerospace Science and Technology*, *55*, 431–438.

Jarry, G., Couellan, N., & Delahaye, D. (2019). On the use of generative adversarial networks for aircraft trajectory generation and atypical approach detection. In *Proceedings of the 6th ENRI International Workshop on ATM/CNS*.

Jarry, G., Hassoumi, A., Delahaye, D., & Hurter, C. (2020). Interactive trajectory modification and generation with FPCA. In *Proceedings of the 9th international conference for research in air transportation*.

Karaletsos, T., Belongie, S., & Rätsch, G. (2015). Bayesian representation learning with oracle constraints. arXiv preprint arXiv:1506.05011.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

Kingma, D. P., & Welling, M. (2019). An introduction to variational autoencoders. arXiv preprint arXiv:1906.02691.

Koyuncu, E., Uzun, M., & Inalhan, G. (2016). Cross-entropy-based cost-efficient 4D trajectory generation for airborne conflict resolution. *Proceedings of the Institution of Mechanical Engineers, Part G (Journal of Aerospace Engineering)*, *230*(9), 1605–1631.

Krauth, T., Morio, J., Olive, X., Figuet, B., & Monstein, R. (2021). Synthetic aircraft trajectories generated with multivariate density models. *Engineering Proceedings*, *13*(1), 7.

Lazzara, M., Chevalier, M., Colombo, M., Garcia, J. G., Lapeyre, C., & Teste, O. (2022). Surrogate modelling for an aircraft dynamic landing loads simulation using an LSTM AutoEncoder-based dimensionality reduction approach. *Aerospace Science and Technology*, *126*, Article 107629.

Lenz, S., Hess, M., & Binder, H. (2021). Deep generative models in DataSHIELD. *BMC Medical Research Methodology*, *21*(1), 1–16.

Monstein, R., Figuet, B., Krauth, T., Waltert, M., & Dettling, M. (2022). *Large landing trajectory data set for go-around analysis*. Zenodo, http://dx.doi.org/10.5281/zenodo.7148117, [This research was funded by the Swiss Federal Office of Civil Aviation grant number SFLV 2018-037].

Murça, M. C. R., & de Oliveira, M. (2020). A data-driven probabilistic trajectory model for predicting and simulating terminal airspace operations. In *Proceedings of the 39th IEEE/AIAA digital avionics systems conference (DASC)*.

Olive, X. (2019). Traffic, a toolbox for processing and analysing air traffic data. *Journal of Open Source Software*, *4*(39).

Olive, X., Basora, L., Viry, B., & Alligier, R. (2020). Deep trajectory clustering with autoencoders. In *Proceedings of the 9th international conference for research in air transportation*.

Olive, X., Sun, J., Murca, M. C. R., & Krauth, T. (2021). A framework to evaluate aircraft trajectory generation methods. In *Proceedings of the 14th USA/Europe air traffic management research and development seminar*.

Schäfer, M., Strohmeier, M., Lenders, V., Martinovic, I., & Wilhelm, M. (2014). Bringing up OpenSky: A large-scale ADS-B sensor network for research. In *IPSN-14 proceedings of the 13th international symposium on information processing in sensor networks* (pp. 83–94). IEEE.

Székely, G. J., Rizzo, M. L., et al. (2004). Testing for equal distributions in high dimension. *InterStat*, *5*(16.10), 1249–1272.

Tokdar, S. T., & Kass, R. E. (2010). Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, *2*(1), 54–60.

Tomczak, J., & Welling, M. (2018). VAE with a VampPrior. In *International conference on artificial intelligence and statistics* (pp. 1214–1223). PMLR.

Vondrick, C., Pirsiavash, H., & Torralba, A. (2016). Generating videos with scene dynamics. *Advances in Neural Information Processing Systems*, *29*.

Zhang, W., Hu, M., & Du, J. (2022). An end-to-end framework for flight trajectory data analysis based on deep autoencoder network. *Aerospace Science and Technology*, *127*, Article 107726.