1 Introduction

Air Traffic Management is facing several challenges. The growth of the traffic is going to continue after the pandemic hiatus, while the social pressure to reduce the environmental impact of the aviation sector is rising. In order to build a more efficient and environment-friendly air traffic control (ATC) system while improving the level of safety; automation, and in particular machine learning, will be a central tool in the next decades. However, the ATC is a critical system by nature and new automated tools must not only be trustworthy but also certified. Even for decision-support tools which are less critical and do not need to be certified, air traffic controllers have shown limited trust to automated systems and prefer to rely on classical methods which are more trustworthy but less efficient. In practice, ATC procedures have not evolved much for the last fifty years. Thus, there is an urgent need in the ATC sector to produce robust and trustworthy machine learning systems.

For the last few years, the machine learning community have started to study the robustness problems of machine learning models, and of neural networks in particular. This study was motivated by the high sensitivity of neural networks to adversarial attacks (small perturbations that are able to fool a network). Other issues linked to robustness are sensitivity to noise, ability to detect out-of-distribution data, and quantification of the uncertainty in order to avoid model overconfidence. Uncertainty is traditionally divided into two types: aleatoric and epistemic. Aleatoric uncertainty (also called data uncertainty) is intrinsic to the task at hand while epistemic uncertainty (also called model uncertainty) represents the model ignorance and can be reduced with more training data. A third type of uncertainty can be introduced as the uncertainty over the distribution that generated the current sample. Two approaches have been developed to address the robustness of neural networks: the Bayesian approach and the probabilistic approach.

The Bayesian approach consists of seeing the parameters of the model as probability distributions instead of point estimates. This requires to choose a prior distribution for the parameters, then to compute a Bayesian inference in order to obtain the posterior distribution of the parameters conditioned on the training data. Since the exact Bayesian inference is intractable, this approach relies on approximation techniques such as Variational Inference, Markov Chain Monte Carlo, expectation propagation, or Laplace approximation, that enable to train the distributions of the parameters using backpropagation. Examples of Bayesian methods for neural networks uncertainty quantification include: MC-dropout, Bayes By Backpropagation (BBP), preconditioned stochastic gradient Langevin dynamics (p-SGLD), and extended variational density propagation (exVDP).

The probabilistic approach regroups several methods that try to follow probability distributions modeling uncertainty as they propagate through the network. This approach takes its roots in robust optimization. Examples of such methods are: prior network and SDE-Net. We can also include the ensembling methods that train multiple networks and use the variance of their predictions to quantify the uncertainty. This includes Deep Ensemble and particle optimization.

Here, we will pursue a third approach that avoid the difficulty of tracking probability distributions along highly non-linear transformations by keeping only the geometrical aspect of the probability distributions. This approach is based on information geometry. We assume that the parameters of the model (weights and biases) are fixed and we focus on the relation between the input and the output. The output space is endowed with a family of probability distributions equipped with the Fisher information metric (FIM), making it a Riemannian manifold. This metric is then pulled back to the input space where it induces a pseudo-metric (if the input space has a lower dimension than the output space). The kernel of this pseudo-metric defines a foliation whose leaves are neutral submanifolds for the metric and thus, represents the link between the input space and the output space through the effect of the network.

The typical objects used in ATC are trajectories, flight plans, flows etc. In other words, ATC systems work with time series. More specifically, we will focus on the prediction of trajectories or the prediction of traffic complexity which is by nature a regression task over time series. The canonical models to treat such data in deep learning are recurrent neural networks (LSTM, GRU) and transformers. In this work, we study models that predict such time series. The aleatoric uncertainty is modeled by seeing the output as a random variable. The probability distribution of this random variable is assumed to belong to a family of distributions from which we can perform the analysis described in the previous paragraph.

2 General framework

In this section, following [1], we present how to use geometry for the study of neural networks robustness.

2.1 Parameterized families of distributions

Let $N(x,\omega)$ be a neural network where $x \in \mathbb{R}^n$ is the input and $\omega \in \mathbb{R}^p$ are the parameters (weights and biases). The output of this neural network is the parameter $\theta \in \mathbb{R}^q$ of a distribution $p(y|\theta(x,\omega))$ where the parameter $\theta = \theta(x,\omega) \in \mathbb{R}^q$ is a smooth function of x and ω. From now on, we assume that $p(y|\theta(x,\omega))$ belongs to an *exponential family*. Let us briefly define what is an exponential family. Let P be a measure over $\mathbb{R}^m$ and assume that $y|\theta$ is absolutely continuous with respect to P. The density of $y|\theta$, denoted by $p(y|\theta(x,\omega)))$ is

$$p(y|\theta(x,\omega)) = \exp\left(\eta(\theta)^T t(y) - \psi(\theta)\right), \tag{1}$$

where t is a sufficient statistic, $\eta(\theta)$ is called the natural parameter and ψ is chosen such that

$$\int \exp\left(\eta(\theta)^T t(y) - \psi(\theta)\right) dP(y) = 1$$

Furthermore, we assume that the support of $p(y|\theta(x,\omega))$ does not depend on x, ω. Given a function $g(x,y)$, the Leibniz integral rule tells us that

$$\frac{\partial}{\partial x^i}\int_{a(x)}^{b(x)} g(x,y)dy = g(x,b(x))\frac{\partial}{\partial x^i}b(x) - g(x,a(x))\frac{\partial}{\partial x^i}a(x) + \int_{a(x)}^{b(x)} \frac{\partial}{\partial x^i}g(x,y)dy.$$

Since the support of $p(y|\theta(x,\omega))$ does not depend on x, ω, the limits of integration do not depend on x, ω either and we can interchange integration with respect to P and partial derivative.

We give two examples of exponential families.

- For a classification task, $m = 1$ and P is the counting measure over $1, \dots, q$. We have[1] $t(y) = (\delta_1(y), \dots, \delta_q(y))^T$, $\eta(\theta) = (\ln\theta^1, \dots, \ln\theta^q)^T$, and $\psi(\theta) = 0$. Hence

  $$p(y|\theta(x,\omega)) = \prod_{i=1}^{q} (\theta^i)^{\delta_i(y)}.$$

  The parameter $\theta^i(x,\omega)$ is the probability that $y|\theta$ belongs to the i-th class, which is generally obtained with $\theta^i(x,\omega) = \mathrm{softmax}(s(x,\omega))_i$ for some score function s. Thus $N(x,\omega) = \mathrm{softmax}(s(x,\omega))$.

- For a regression task, P is the Lebesgue measure and the parameter $\theta = (\mu, \Sigma)$ is the mean and covariance matrix of a multivariate normal distribution. We have $t(y) = (y, yy^T)^T$, $\eta(\mu,\Sigma) = (\Sigma^{-1}\mu, \Sigma^{-1}/2)^T$, and $\psi(\mu,\Sigma) = \frac{1}{2}(\mu^T\Sigma^{-1}\mu + \ln\det\Sigma + m\ln 2\pi)$. Hence

  $$p(y|\mu(x,\omega),\Sigma(x,\omega)) = \frac{1}{\sqrt{(2\pi)^m \det\Sigma}} \exp\left(-\frac{1}{2}(y-\mu)^T\Sigma^{-1}(y-\mu)\right).$$

  Thus $N(x,\omega) = (\mu(x,\omega), \Sigma(x,\omega))$.

Let us list all the assumptions for the rest of this document:

1. The random variable $y|\theta(x,\omega)$ belongs to an exponential family where P is the Lebesgue measure.
2. The support of $p(y|\theta(x,\omega))$ does not depend on x, ω and we can interchange integration and partial derivative.
3. The moments of $\frac{\partial}{\partial\theta^i}\ln p(y|\theta(x,\omega))$ exist up to necessary orders.
4. For any fixed x, ω, the q functions $y \mapsto \frac{\partial}{\partial\theta^i}\ln p(y|\theta(x,\omega))$ are linearly independent.

[1] where $\delta_i(y) = 1$ if $y = i$ and 0 otherwise.

2.2 The local data matrix

Let $q(y|x)$ be the true distribution of y (the "labels"). In order to train the network N, we use the *cross entropy loss function*

$$L(x, \omega) = -\mathbb{E}_{y\sim q}[\ln p(y|\theta(x, \omega))], \tag{2}$$

which is the sum of the Kullback-Leibler divergence between q and p with the differential entropy of the distribution $q(y|x)$ denoted $H(q(y|x))$:

$$\begin{aligned} L(x, \omega) &= \mathbb{E}_{y\sim q}[-\ln p(y|\theta(x, \omega))] \\ &= \mathbb{E}_{y\sim q}\left[\ln \frac{q(y|x)}{p(y|\theta(x, \omega))}\right] - \mathbb{E}_{y\sim q}[\ln q(y|x)] \\ &= \mathrm{KL}(q(y|x)||p(y|\theta(x, \omega))) + H(q(y|x)). \end{aligned}$$

Note that $H(q(y|x))$ does not depend on ω and does not affect the training.

Now, let $x_0 \in \mathbb{R}^n$, $\omega_0 \in \mathbb{R}^p$ be fixed and consider the function $F : \mathbb{R}^n \to \mathbb{R}$ defined by

$$F(x) = \mathrm{KL}(p(y|\theta(x_0, \omega_0))||p(y|\theta(x, \omega_0))). \tag{3}$$

Let $\delta x = x - x_0$. The second-order Taylor approximation of F at x_0 is

$$F(x) = F(x_0) + (\nabla_x F(x_0))^T \delta x + (\delta x)^T \nabla_{xx} F(x_0) \delta x + o\left(\|\delta x\|^2\right)$$

We have $F(x_0) = 0$ and

$$\begin{aligned} \nabla_x F(x) &= \nabla_x \int p(y|\theta(x_0, \omega_0)) \ln \frac{p(y|\theta(x_0, \omega_0))}{p(y|\theta(x, \omega_0))} dy \\ &= -\nabla_x \int p(y|\theta(x_0, \omega_0)) \ln p(y|\theta(x, \omega_0)) dy \\ &= -\int p(y|\theta(x_0, \omega_0)) \nabla_x \ln p(y|\theta(x, \omega_0)) dy \\ &= -\int p(y|\theta(x_0, \omega_0)) \frac{\nabla_x p(y|\theta(x, \omega_0))}{p(y|\theta(x, \omega_0))} dy. \end{aligned}$$

For $x = x_0$, we get

$$\begin{aligned} \nabla_x F(x_0) &= -\int \nabla_x p(y|\theta(x_0, \omega_0)) dy \\ &= -\nabla_x \int p(y|\theta(x_0, \omega_0)) dy \\ &= -\nabla_x 1 \\ &= 0. \end{aligned}$$

Now, we consider the Hessian matrix of F:

$$\nabla_{xx} F(x) = -\int p(y|\theta(x_0, \omega_0)) \nabla_{xx} \ln p(y|\theta(x, \omega_0)) dy.$$

Let $1 \leq i, j \leq n$ and denote the partial derivative by $\partial/\partial x^i = \partial_i$. We have

$$\begin{aligned}
(\nabla_{xx} F(x_0))_{ij} &= -\int p(y|\theta(x_0,\omega_0))\partial_i\partial_j \ln p(y|\theta(x_0,\omega_0))dy \\
&= \partial_j F(x_0) - \int p(y|\theta(x_0,\omega_0))\partial_i\partial_j \ln p(y|\theta(x_0,\omega_0))dy && \text{(since } \partial_j F(x_0) = 0\text{)} \\
&= \int p(y|\theta(x_0,\omega_0))\partial_j \ln p(y|\theta(x_0,\omega_0))dy - \int p(y|\theta(x_0,\omega_0))\partial_i\partial_j \ln p(y|\theta(x_0,\omega_0))dy \\
&= \int \partial_i p(y|\theta(x_0,\omega_0))\partial_j \ln p(y|\theta(x_0,\omega_0))dy && \text{(using integration by part)} \\
&= \int p(y|\theta(x_0,\omega_0))\partial_i \ln p(y|\theta(x_0,\omega_0))\partial_j \ln p(y|\theta(x_0,\omega_0))dy \\
&= \mathbb{E}_{y\sim(x_0,\omega_0)}[\partial_i \ln p(y|\theta(x_0,\omega_0))\partial_j \ln p(y|\theta(x_0,\omega_0))].
\end{aligned}$$

Define the local data matrix[2] $G(x,\omega)$ by

$$G(x,\omega) = \mathbb{E}_{y\sim(x,\omega)}[\nabla_x \ln p(y|\theta(x,\omega))(\nabla_x \ln p(y|\theta(x,\omega)))^T]. \tag{4}$$

Finally, for any x and any deviation δx from x, we obtain

$$\mathrm{KL}(p(y|\theta(x,\omega))||p(y|\theta(x+\delta x,\omega))) = (\delta x)^T G(x,\omega)(\delta x) + o\left(\|\delta x\|^2\right). \tag{5}$$

Now, we state some properties of the local data matrix $G(x,\omega)$.

Proposition 2.1. *Let $x \in \mathbb{R}^n$ and $\omega \in \mathbb{R}^p$.*

1. *$G(x,\omega)$ is a positive semidefinite symmetric matrix.*
2. *$\ker G(x,\omega) = \left(\mathrm{span}\{\nabla_x \theta^i(x,\omega)\}\right)^\perp$, where $\perp$ means the orthogonal space with respect to the canonical dot product of $\mathbb{R}^n$.*
3. *$\mathrm{rank}\ G(x,\omega) \leq q$.*

Proof. This is Proposition 2.1 of [1].

1. $G(x,\omega)$ is symmetric because

$$\begin{aligned}
G(x,\omega)_{ij} &= \mathbb{E}_{y\sim(x,\omega)}[\partial_i \ln p(y|\theta(x,\omega))\partial_j \ln p(y|\theta(x,\omega))] \\
&= \mathbb{E}_{y\sim(x,\omega)}[\partial_j \ln p(y|\theta(x,\omega))\partial_i \ln p(y|\theta(x,\omega))] \\
&= G(x,\omega)_{ji}.
\end{aligned}$$

   Let $u \in \mathbb{R}^n$. We have

$$\begin{aligned}
u^T G(x,\omega) u &= \mathbb{E}_{y\sim(x,\omega)}[u^T \nabla_x \ln p(y|\theta(x,\omega))(\nabla_x \ln p(y|\theta(x,\omega)))^T u] \\
&= \mathbb{E}_{y\sim(x,\omega)}[((\nabla_x \ln p(y|\theta(x,\omega)))^T u)^2] \geq 0,
\end{aligned}$$

   hence $G(x,\omega)$ is positive semidefinite.

2. First, we show that $\left(\mathrm{span}\{\nabla_x \theta^i(x,\omega)\}\right)^\perp \subset \ker G(x,\omega)$. Using the chain rule:

$$\nabla_x \ln p(y|\theta(x,\omega)) = \sum_{i=1}^{q} \frac{\partial}{\partial \theta^i} \ln p(y|\theta) \nabla_x \theta^i(x,\omega),$$

   hence, for all y, $\nabla_x \ln p(y|\theta(x,\omega)) \in \mathrm{span}\{\nabla_x \theta^i(x,\omega)\}$. Let $u \in \left(\mathrm{span}\{\nabla_x \theta^i(x,\omega)\}\right)^\perp$. We have

$$G(x,\omega)u = \mathbb{E}_{y\sim(x,\omega)}[\nabla_x \ln p(y|\theta(x,\omega))(\nabla_x \ln p(y|\theta(x,\omega)))^T u] = 0,$$

[2] We follow [1] in naming $G(x,\omega)$ the local data matrix to distinguish it from the local Fisher matrix for which the derivatives are taken with respect to ω instead of x.

since u and $\nabla_x \ln p(y|\theta(x,\omega))$ are orthogonal. Thus $u \in \ker G(x,\omega)$.

Now, let $u \in \ker G(x,\omega)$. Then

$$\begin{aligned} u^T G(x,\omega) u &= 0 \\ &= \mathbb{E}_{y\sim(x,\omega)}[u^T \nabla_x \ln p(y|\theta(x,\omega))(\nabla_x \ln p(y|\theta(x,\omega)))^T u] \\ &= \mathbb{E}_{y\sim(x,\omega)}[((\nabla_x \ln p(y|\theta(x,\omega)))^T u)^2], \end{aligned}$$

Hence, for almost all y,

$$\sum_{i=1}^{q} \frac{\partial}{\partial \theta^i} \ln p(y|\theta)(\nabla_x \theta^i(x,\omega))^T u = 0.$$

Since the functions $y \mapsto \frac{\partial}{\partial \theta^i} \ln p(y|\theta)$ are linearly independent, we have for all i:

$$(\nabla_x \theta^i(x,\omega))^T u = 0.$$

Hence, $u \in \left(\text{span}\{\nabla_x \theta^i(x,\omega)\}\right)^\perp$.

3. According to the preceding point, rank $G(x,\omega) = n - \dim(\ker G(x,\omega)) = n - \dim\left(\left(\text{span}\{\nabla_x \theta^i(x,\omega)\}\right)^\perp\right)$.
   Moreover, $\dim\left(\text{span}\{\nabla_x \theta^i(x,\omega)\}\right) \leq q$ thus $\dim\left(\left(\text{span}\{\nabla_x \theta^i(x,\omega)\}\right)^\perp\right) \geq n - q$.
   Hence rank $G(x,\omega) \leq q$.

□

The local data matrix can be obtained as the pullback of the Fisher information metric (FIM) on the output space $\mathbb{R}^q$. The pullback can be seen as "retropropagating" the geometry from the output space $\mathbb{R}^q$ to the input space $\mathbb{R}^n$. Denote by $\mathcal{H} = \{p(y|\theta)\}$ the family parameterized by $\theta \in \mathbb{R}^q$. The family $\mathcal{H}$ is a differentiable manifold and the parameter θ can be seen as a map $\theta : p(y|\theta) \in \mathcal{H} \mapsto \theta \in \mathbb{R}^q$ which is a coordinate system of $\mathcal{H}$. Hence, we can identify $\mathcal{H}$ and $\mathbb{R}^q$. We can see our model as a map $N : (x,\omega) \in \mathbb{R}^n \times \mathbb{R}^p \mapsto N(x,\omega) = \theta \in \mathbb{R}^q$. The FIM on $\mathcal{H}$ is a Riemannian metric defined in the coordinates θ by

$$G_{out}(\theta) = \mathbb{E}_{y\sim\theta}[\nabla_\theta \ln p(y|\theta)(\nabla_\theta \ln p(y|\theta))^T].$$

Then, the local data matrix can be obtained with

$$G(x,\omega) = N^* G_{out}(N(x,\omega)),$$

where N^* is the pullback of N. We use Greek symbols on $\mathcal{H}$ (α, β etc.) and Latin symbols on $\mathbb{R}^n$ (i, j etc.), and we use the notations $\partial_\alpha = \partial/\partial\theta^\alpha$ and $\partial_i = \partial/\partial x^i$. In coordinate, we get

$$\begin{aligned} G(x,\omega)_{ij} &= G(x,\omega;\partial_i,\partial_j) \\ &= G_{out}(N(x,\omega); N_*\partial_i, N_*\partial_j) \\ &= G_{out}(N(x,\omega); \frac{\partial\theta^\alpha \circ N}{\partial x^i}\partial_\alpha, \frac{\partial\theta^\beta \circ N}{\partial x^j}\partial_\beta) \\ &= \frac{\partial\theta^\alpha \circ N}{\partial x^i}\frac{\partial\theta^\beta \circ N}{\partial x^j} G_{out}(N(x,\omega); \partial_\alpha, \partial_\beta) \\ &= J_i^\alpha(x,\omega) J_j^\beta(x,\omega) G_{out}(N(x,\omega))_{\alpha\beta}, \end{aligned}$$

which can be rewritten

$$G(x,\omega) = J(x,\omega)^T G_{out}(N(x,\omega)) J(x,\omega).$$

where J is the Jacobian matrix of N between the basis $\{\partial_i\}$ and $\{\partial_\alpha\}$.

2.3 The image and kernel foliations

Equation 5 tells us that if we move along the directions of $\ker G(x,\omega)$ in the input space, the probability distribution $p(y|\theta(x,\omega))$ is constant (this is true for a small enough displacement). For these directions to exist, a sufficient condition is that the dimension of the input space n is larger than the dimension of the output space q, which is typically verified in supervised learning. According to [1], the directions of $\ker G(x,\omega)$ are the *noise directions*, i.e., directions in which the inputs get more and more noisy while the network still predict the same output with the same confidence. On the other hand, moving along directions in $(\ker G(x,\omega))^\perp$ change the output distribution $p(y|\theta(x,\omega))$, the model will predict different outputs with high confidence. These are the privileged directions for adversarial attacks.

Assume that $G(x,\omega)$ has constant rank with respect to x (remember that ω is fixed). This can be verified experimentally.

Remark 2.2. However, I do not know what are the conditions for the local data matrix to have constant rank. According to [1] and other papers, this is true only for non fully trained models (the rank seems to shrink when the model weights reach a local optimum) but I do not have any explanation for this phenomenon.

We can define a *distribution* D by

$$x \in \mathbb{R}^n \mapsto D_x = \ker G(x,\omega) \subset \mathbb{R}^n.$$

A distribution consists in associating to each point x of a manifold $\mathcal{M}$ a subspace of the tangent space T_xM in a smooth way. Moreover, we ask the corresponding subspaces to have constant dimension. Here, we assume that, for all x, we have $\dim \ker G(x,\omega) = r$. For each point $x \in \mathbb{R}^n$, we would like to find a submanifold $N_x \subset \mathbb{R}^n$ such that $x \in N_x$ and for any $x' \in N_x$, we have $T_{x'}N_x = D_{x'}$. If such N_x uniquely exists for all $x \in \mathbb{R}^n$, then the distribution D is said to be *integrable*. The submanifold N_x is called the *leaf* of x and the set of all leaves for all $x \in \mathbb{R}^n$ is called a *foliation*. The entire manifold $\mathbb{R}^n$ is thus the disjoint union of all the leaves of the foliation. Since our distribution D is defined by the kernel of the local data matrix, we call it the *kernel distribution*, its associated foliation the *kernel foliation*, and its leaves the *kernel leaves*. Moving along a kernel leaf means moving along a kernel direction at each point. Hence, the output distribution does not change along a kernel leaf.

A necessary and sufficient condition for a distribution to be integrable is given by Frobenius Theorem. First, we define the notion of *involutive* distribution.

Definition 2.3 (Involutive property)**.** Let D be a distribution and let X and Y be smooth vector fields belonging to D (i.e., for all x, $X_x \in D_x$ and $Y_x \in D_x$). Define the Lie bracket of vector fields by $[X,Y]_x = X_xY - Y_xX$.
Then D is said to be involutive if $[X,Y] \in D$.

Then we can state Frobenius Theorem.

Theorem 2.4 (Frobenius)**.** *A distribution D is integrable if and only if it is involutive.*

There is no reason for an arbitrary neural network to induce an involutive kernel distribution. In [1], the authors show that for a classifier using ReLU activation for the hidden layers and softmax for the last layer, the kernel distribution is involutive (assuming the local data matrix has constant rank). This result comes from the fact that such a network is piecewise linear. However, the authors also claim that it can be shown experimentally that the kernel distribution induced by the local Fisher matrix (obtained by differentiating with respect to the weight ω instead of the data point x) is not involutive. Then, the authors focus on the *transverse foliation* of the kernel foliation which we call the *data foliation* defined by $x \mapsto (\ker G(x,\omega))^\perp$. They show experimentally that all the training dataset lie on a unique leaf of the image foliation. The advantage of the image leaves compared to the kernel leaves is that it is possible to use $G(x,\omega)$ to define a Riemannian metric on each image leaf, since by definition, the local data matrix is non-degenerate on $(\ker G(x,\omega))^\perp$.

Remark 2.5. I do not know what are the conditions for the kernel distribution to be involutive. For the data x, we know it is true for piecewise linear network with softmax but I do not understand the intuition behind this result. Moreover, we know it is not true for the parameters ω, but once again I wonder what is the intuition behind it.

2.4 Sequential models

In this paragraph, we describe our framework for the prediction of time series.

Let S be an open set of $\mathbb{R}^d$ called the *state space*. Let $\phi : S \to S$ be a diffeomorphism of S. We see ϕ as a dynamical system such that $s_{t+1} = \phi(s_t)$. Let $f : S \to \mathbb{R}$ be a smooth function called the *measurement function*. Let $n \geq 2d + 1$. Consider the map $\Psi : s \in S \mapsto (f(s), f(\phi(s)), \ldots, f(\phi^{n-1}(s)))^T \in \mathbb{R}^n$. We have the following result.

Theorem 2.6 (Takens)**.** *For generic ϕ and f, Ψ is an embedding.*
In other words, there exist a subset $\tilde{S} \subset \mathbb{R}^n$ and a dynamical system $\tilde{\phi} : \tilde{S} \to \tilde{S}$ which is conjugated with ϕ, i.e., $\phi = \Psi^{-1} \circ \tilde{\phi} \circ \Psi$ for some diffeomorphism $\Psi : S \to \tilde{S}$.

We can see the input space $\tilde{S} \subset \mathbb{R}^n$ as a set of time series of length n. Given a time series $x \in \tilde{S} \subset \mathbb{R}^n$, the goal of the model N is to predict a time series $y \in \mathbb{R}^m$ of length m. The simplest example is to use $x = (f(s), f(\phi(s)), \ldots, f(\phi^{n-1}(s)))^T$ to predict the next measurement $y = f(\phi^n(s))$ (here we have $m = 1$) or a sequence of next measurements $y = (f(\phi^n(s)), f(\phi^{n+1}(s)), \ldots, f(\phi^{n+m-1}(s)))^T$ for arbitrary m. In both examples, the goal of the model is to learn the conjugate dynamic $\tilde{\phi}$. In the general case, we assume that there is a smooth function $g : \mathbb{R}^n \to \mathbb{R}$ such that $y^i = g(\tilde{\phi}^i(x))$ for $1 \leq i \leq m$. In the examples, g was the projection to the last component. We call a model whose objective is to learn this mapping, i.e., $N(x, \omega) \approx (g(\tilde{\phi}(x)), \ldots, g(\tilde{\phi}^m(x)))^T$, a *sequential model*.

Our goal is to study the kernel foliation of a sequential model N in order to guarantee the robustness of such models to noise, adversarial attacks, and out-of-distribution samples. To apply our framework, we must see $y \in \mathbb{R}^m$ as a random variable whose distribution is parameterized by the model output $N(x, \omega) \in \mathbb{R}^q$. Hence, N does not infer y directly, but only the parameter of the distribution of y. In the following, we assume that this distribution belongs to the family $\mathcal{H}_m$ of m-dimensional normal distributions, hence $q = m + \frac{m(m+1)}{2} = \frac{m(m+3)}{2}$. A sufficient condition for the existence of the kernel foliation is $n > q$, i.e., $m < (\sqrt{9 + 8n} - 3)/2$ and we assume that this condition is met in the following. Once equipped with the Fisher information metric G_{out}, $\mathcal{H}_m$ becomes a q-dimensional Riemannian manifold.

2.5 Linear stochastic differential equation

In order to compute the local data matrix $G(x, \omega)$, we have to compute the metric $G_{out}(\theta)$. In practice, most sequential models infer the output y element by element. For example, recurrent neural networks predict y^1, then y^2, and so on until y^m. In our framework, this means that the model infers a mean μ_i and a standard deviation σ_i for each $1 \leq i \leq m$ but is unable to infer the covariance terms of the m-dimensional normal distribution. These terms are generally assumed to be zero, which is equivalent to assume that the elements of y are independent. However, since y is seen as a time series, its elements have no reason to be independent, and the covariance between them contains a lot of information about the dynamic of the predicted time series. Hence, we need to know the true distribution of y, or at least to approximate it. The distribution of y depends on N which can be highly non-linear. In this paragraph, we describe a simple model that is able to produce m-dimensional normal distributions whose elements exhibit a dynamical relationship. In the next paragraph, we see how to link this simpler case with the true distribution of y.

To approximate the true distribution of y, the idea is to see y as a sequence of samples of a stochastic process which is the solution of a linear stochastic differential equation. This can be interpreted as a "linearization" of the model. Consider the following linear scalar autonomous stochastic differential equation (SDE) with additive noise

$$\begin{aligned} dy(s) &= (a.y(s) + b)dt + c.dw(s), \\ y(0) &= y_0 \text{ p.s.} \end{aligned}$$

where $y : \mathbb{R}^+ \times (\Omega, \mathcal{F}, \mathbb{P}) \to (\mathbb{R}, \mathcal{B}(\mathbb{R}))$ is a stochastic process, w is a standard Wiener process, and $a, b, c, y_0 \in \mathbb{R}$. Assume that $a \neq 0$. This SDE is a Langevin equation and its solution is an Ornstein-Uhlenbeck (OU)

process. This process is stationary, Gaussian, Markov and continuous in probability. One has:

$$\mu(s) = -\frac{b}{a} + \left(x + \frac{b}{a}\right) e^{as},$$
$$\sigma^2(s) = \frac{c^2}{2a}(e^{2as} - 1).$$

Let $\tau > 0$. For integers t between 1 and m, define

$$y_t = y(t.\tau).$$

We want to derive the distribution of $y = (y_1, \ldots, y_m)$. Since the OU process is Gaussian, the distribution of any sample is a Gaussian vector.

Remark 2.7. Is the converse true? Is any Gaussian vector a sample of an OU process? If not, the set of distributions that are a sample of an OU process is a submanifold of $\mathcal{H}_m$. We leave this question aside for now.

Let $\mathcal{N}_m$ be the submanifold of $\mathcal{H}_m$ composed of the distributions that are samples of an OU process (possibly, we have $\mathcal{N}_m = \mathcal{H}_m$). To compute the FIM $G_{OU}(\theta)$ on $\mathcal{N}_m$, we need to compute the FIM $G_{out}(\theta)$ on $\mathcal{H}_m$. Then $G_{OU}(\theta)$ is the metric induced by $G_{out}(\theta)$ on $\mathcal{N}_m$. Let $\mathcal{S}_m$ be the set of positive definite symmetric matrices of dimension $m \times m$.

Proposition 2.8. *If $\theta = (\mu, \Sigma)$, where $\mu \in \mathbb{R}^m$ is the mean and $\Sigma \in \mathcal{S}_m$ is the covariance matrix, then*

$$G_{out}(\mu, \Sigma) = \frac{1}{2}\mathrm{tr}((\Sigma^{-1}d\Sigma)^2) + d\mu^T\Sigma^{-1}d\mu.$$

The proof is incomplete.

Proof. The log-likelihood is

$$\ln p(y|\mu, \Sigma) = -\frac{1}{2}(y-\mu)^T\Sigma^{-1}(y-\mu) - \frac{1}{2}\ln(\det\Sigma) - \frac{m}{2}\ln 2\pi.$$

The basis of vector fields induced by the coordinates (μ, Σ) can be identified as follows

$$\frac{\partial}{\partial\mu^i} \leftrightarrow e^i,$$
$$\frac{\partial}{\partial\sigma^{ij}} \leftrightarrow E^{ij},$$

where the i-th component of e^i is 1 and the others are 0, the (i,j)-th and (j,i)-th components of E^{ij} are 1 and the others are 0 (hence if $i = j$, only the (i,i)-th component is equal to 1). In other words, $\{e^i\}_{1\le i\le m}$ is the canonical basis of $\mathbb{R}^m$ and $\{E^{ij}\}_{1\le i\le j\le m}$ is the canonical basis of the $m \times m$ symmetric matrices.

Let σ^{ij} be the components of Σ and σ_{ij} be the components of Σ^{-1}. Hence, using Einstein summation convention, $\sigma^{ik}\sigma_{kj} = \delta^i_j$. One has

$$\frac{\partial \ln p(y|\mu,\Sigma)}{\partial\mu^i} = (y^k - \mu^k)\sigma_{ik}.$$

Then

$$\frac{\partial^2 \ln p(y|\mu,\Sigma)}{\partial\mu^i\partial\mu^j} = -\sigma_{ij}.$$

Hence

$$\begin{aligned} G_{out}(\mu, \Sigma; e^i, e^j) &= -\mathbb{E}\left[\frac{\partial^2 \ln p(y|\mu,\Sigma)}{\partial\mu^i\partial\mu^j}\right], \\ &= \sigma_{ij}, \\ &= e^i\Sigma^{-1}e^j. \end{aligned}$$

Now, by differentiating the relation $\Sigma\Sigma^{-1} = I$ with respect to σ^{ij}, we obtain $(\partial\Sigma/\partial\sigma^{ij})\Sigma^{-1}+\Sigma(\partial\Sigma^{-1}/\partial\sigma^{ij}) = 0$, thus $\partial\Sigma^{-1}/\sigma^{ij} = -\Sigma^{-1}(\partial\Sigma/\sigma^{ij})\Sigma^{-1} = (-\sigma_{ki}\sigma_{lj})_{(k,l)}$. Since $\partial l(y,(\mu,\Sigma))/\partial\mu = \Sigma^{-1}(y-\mu)$, one has

$$\frac{\partial^2 \ln p(y|\mu,\Sigma)}{\partial\mu\partial\sigma^{jk}} = -\Sigma^{-2}(y-\mu).$$

Thus

$$G_{out}(\mu,\Sigma;e^i,E^{jk}) = 0.$$

The Jacobi's formula yields $\partial(\det\Sigma)/\partial\sigma^{ij} = c^{ij}$ where $C = (c^{ij})$ is the cofactor matrix of Σ. Hence

$$\frac{\partial \ln p(y|\mu,\Sigma)}{\partial\sigma^{ij}} = \frac{1}{2}(y^a-\mu^a)(y^b-\mu^b)\sigma_{ai}\sigma_{bj} - \frac{1}{2}\frac{c^{ij}}{\det\Sigma}.$$

Then

$$\begin{aligned}\frac{\partial \ln p(y|\mu,\Sigma)}{\partial\sigma^{ij}}\frac{\partial \ln p(y|\mu,\Sigma)}{\partial\sigma^{kl}} = \frac{1}{4}\Bigg[&(y^a-\mu^a)(y^b-\mu^b)(y^c-\mu^c)(y^d-\mu^d)\sigma_{ai}\sigma_{bj}\sigma_{ck}\sigma_{dl}\\ &- \frac{c^{ij}}{\det\Sigma}(y^c-\mu^c)(y^d-\mu^d)\sigma_{ck}\sigma_{dl} - \frac{c^{kl}}{\det\Sigma}(y^a-\mu^a)(y^b-\mu^b)\sigma_{ai}\sigma_{bj} + \frac{c^{ij}c^{kl}}{(\det\Sigma)^2}\Bigg].\end{aligned}$$

Taking expectation

$$\begin{aligned}G_{out}(\mu,\Sigma;E^{ij},E^{kl}) &= \frac{1}{4}\left[\sigma_{ai}\sigma_{bj}\sigma_{ck}\sigma_{dl}(\sigma^{ab}\sigma^{cd}+\sigma^{ac}\sigma^{bd}+\sigma^{ad}\sigma^{bc}) - \frac{c^{ij}}{\det\Sigma}\sigma^{cd}\sigma_{ck}\sigma_{dl} - \frac{c^{kl}}{\det\Sigma}\sigma^{ab}\sigma_{ai}\sigma_{bj} + \frac{c^{ij}c^{kl}}{(\det\Sigma)^2}\right],\\ &= \frac{1}{4}\left[\sigma_{ij}\sigma_{kl}+\sigma_{ik}\sigma_{jl}+\sigma_{il}\sigma_{jk} - \frac{c^{ij}}{\det\Sigma}\sigma_{kl} - \frac{c^{kl}}{\det\Sigma}\sigma_{ij} + \frac{c^{ij}c^{kl}}{(\det\Sigma)^2}\right],\\ &= \frac{1}{4}\left[\sigma_{ik}\sigma_{jl}+\sigma_{il}\sigma_{jk} + \left(\sigma_{ij}-\frac{c^{ij}}{\det\Sigma}\right)\left(\sigma_{kl}-\frac{c^{kl}}{\det\Sigma}\right)\right],\end{aligned}$$

□

Remark 2.9. The first part of the metric, i.e., $\frac{1}{2}\mathrm{tr}((\Sigma^{-1}d\Sigma)^2)$, is somehow a natural metric for the manifold of positive definite matrices. It is also called "DeWitt metric" in physics where it is defined over the manifold of metrics (which are positive definite matrices).

2.6 Approximation of the true output distribution

3 Experiments

3.1 Visualizing the kernel foliation

In order to get insights on the behavior of the kernel foliation, we aim at visualizing it for a low dimensional problem. The dimension of the input space must be at most 3. Since Takens' theorem guarantees the existence of an embedding of the state space with at least $2n+1$ observations, the state space must have dimension $n = 1$.

3.1.1 A simple one-dimensional dynamical system

We are looking for a one-dimensional dynamical system with interesting behaviors to be learn by a neural network. We chose the system

$$\frac{dx}{dt} = \sin(\pi x).$$

This is a non-linear system whose equilibria are the integers, with the odd integers being stable and the even integers being unstable. We chose a time step $\delta t = 2.10^{-2}$ and we create a dataset for a supervised time series prediction such that:

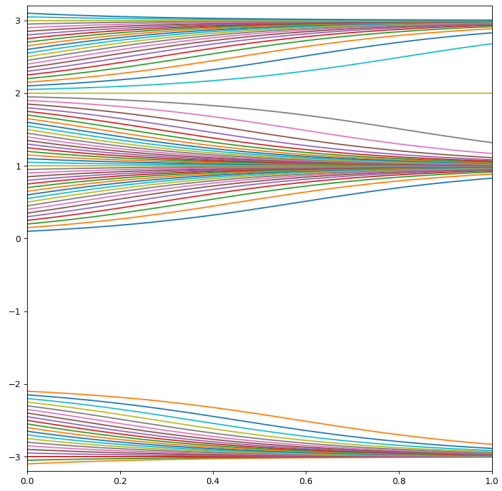


Figure 1: Dataset

- Each input is a vector composed of three successive points $(x(t), x(t+\delta t), x(t+2\delta t))$.
- The corresponding true output is the next point $x(t+3\delta t)$.
- We do not add noise for this experiment.

Figure 1 shows the trajectories used to build the dataset. The initial points are taken between -3 and 3. There is a gap in the data between -2 and 0 in order to see how the model will react to a lack of data. Will it be able to generalize and to infer the existence of the stable equilibria at -1? Will it display a high uncertainty when doing predictions in the interval $[-2, 0]$? Will the kernel foliation reflect the lack of data in this interval? We obtain a dataset of 3936 samples.

3.1.2 A simple neural network

Since this problem is very easy, we use a small neural network. The 3-dimensional input vectors are fed component by component to a LSTM layer with hidden dimension equal to 8. The hidden state of the last LSTM is fed to a fully connected layer that outputs a 2-dimensional vector:

- The first component of this 2-dimensional output vector is $\hat{x}$, the prediction for the true output $x = x(t+3\delta t)$.
- The second component is the standard deviation $\sigma > 0$.

To train this network, we use the negative log-likelihood of the univariate normal distribution as a loss function:

$$L(x, \hat{x}, \sigma) = \ln(\max(\sigma, \epsilon)) + \frac{(x-\hat{x})^2}{2\max(\sigma, \epsilon)^2},$$

where $\epsilon = 10^{-3}$ avoids to divide by 0 or to compute $\ln 0$ and the constant term $\frac{1}{2}\ln 2\pi$ has been discarded.

The model is trained over 500 epochs with the Adam algorithm (learning rate $= 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$). To evaluate the model, we use three test datasets:

- A validation dataset built from 51 trajectories whose initial points are uniformly sampled from the interval $[-3, 3]$.

	Training	Validation	In-distribution	Out-of-distribution
Relative error	0.18	5.72	0.21	15.64
Likelihood	7.29	11.44	6.32	17.87

Table 1: Model evaluation (in %)

- An in-distribution dataset built from 52 trajectories whose initial points are uniformly sampled from $[-3, -2] \cup [0, 3]$.
- An out-of-distribution dataset built from 31 trajectories whose initial points are uniformly sampled from $[-2, 0]$.

We use two evaluation metrics:

- The relative error $r(x, \hat{x}) = \frac{|x-\hat{x}|}{x}$. The error range is $[0, 1]$ and smaller error is better.
- The likelihood defined as the probability of sampling from $\mathcal{N}(\hat{x}, \sigma^2)$ a point that is at least as far from the mean $\hat{x}$ as the true output x. The likelihood range is $[0, 1]$ and higher likelihood is better.

The metrics are averaged over each test datasets. The results are reported in Table 1 using percentages for better readability. The model has very low relative error and low likelihood on the training and in-distribution datasets. This means that the predictions of the model are very close to the true outputs, but the predicted standard deviations are very small such that the model is overconfident even with respect to its good predictions. On the out-of-distribution dataset, we observe a higher relative error associated with a higher likelihood. That means that the model predicts high standard deviations when confronted to out-of-distribution data such that, even if its predictions are poor, its level of confidence is correspondingly low.

To illustrate these remarks, we show in Figure 2a the predicted trajectories and standard deviations along with the true trajectories for several initial points. The trajectories are predicted in a non-recursive mode. By non-recursive mode, we mean that the model uses the last three points of the true trajectory for its next prediction disregarding its own predictions for these three points. We observe that the error is higher on the out-of-distribution range $[-2, 0]$, but that the standard deviation is also higher in this range. Figure 2b shows the same initial points but the predicted trajectories are obtained in a recursive mode, which means that the model uses its own predictions as inputs for the next ones. In order to account for the propagation of error, the standard deviations are accumulated over time (contrary to the non-recursive mode where we show the standard deviation assuming that the error in the inputs is zero).

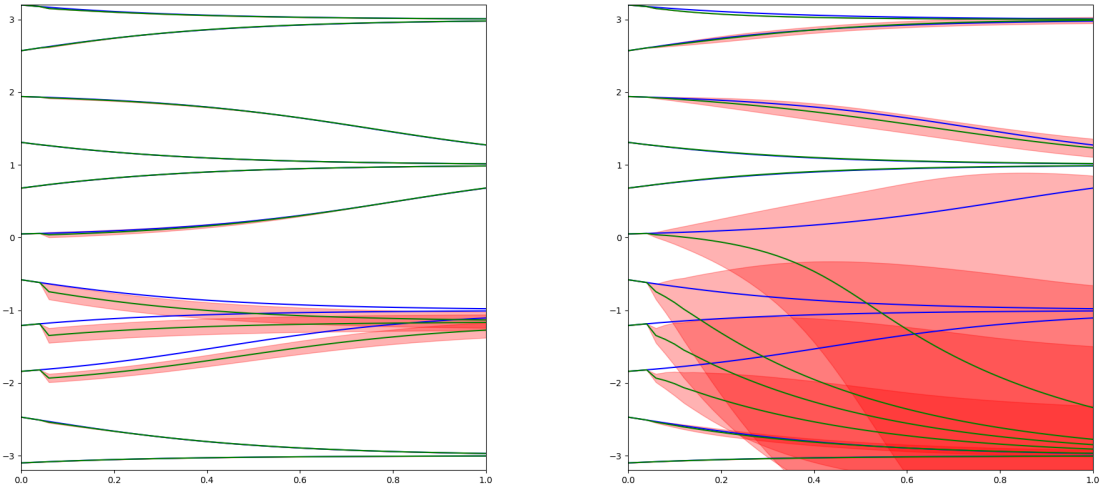
3.1.3 The kernel foliation

Now, we focus on the input space $\mathbb{R}^3$. The state space $\mathbb{R}$ can be embedded in $\mathbb{R}^3$ using the map $x \mapsto (x(0), x(\delta t), x(2\delta t))$ where $x(t) = x$. We obtain the one-dimensional submanifold shown in Figure 3a. The equilibria of the dynamical system correspond to the points of the diagonal $x = y = z$ with integer coordinates. Since, the coordinates of $(x(0), x(\delta t), x(2\delta t))$ are close to each other, all the points of the embedded state space are close to the diagonal. To improve the visualization, we rotate the input space $\mathbb{R}^3$ such that the x-axis is sent to the diagonal $x = y = z$ and we scale the space by $1/\sqrt{3}$ such that the x-axis keeps the same range. The result is shown in Figure 3b where we can see the various equilibria corresponding to points $(x, 0, 0)$.

We use the (mean, standard deviation) coordinates on the output space of univariate normal distribution and the canonical coordinates of $\mathbb{R}^3$ on the input space. Given a point $\theta = (\hat{x}, \sigma)$ of the output space, the corresponding Fisher information matrix is

$$G_{out}(\hat{x}, \sigma) = \begin{pmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{2}{\sigma^2} \end{pmatrix}$$

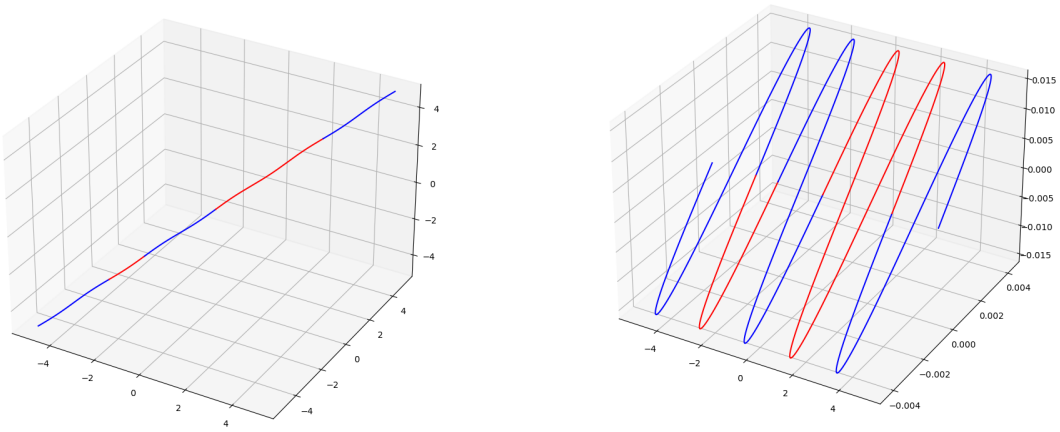
Let $x \in \mathbb{R}^3$ be a point of the input space. Let $J(x)$ be the Jacobian matrix at x between the input space and the output space in the chosen coordinates. Denote by N our model such that $(\hat{x}, \sigma) = N(x)$. Then,



(a) Using non-recursive prediction.

(b) Using recursive prediction.
The standard deviations are accumulated.

Figure 2: True trajectories (blue), predicted trajectories (green), and standard deviations (red).



(a) Using the canonical coordinates.

(b) After a rotation and a scaling. The y- and z-axis ranges have been modified.

Figure 3: Embedding of the state space in the input space. The red parts correspond to the training data while the blue parts were not seen by the model during training.

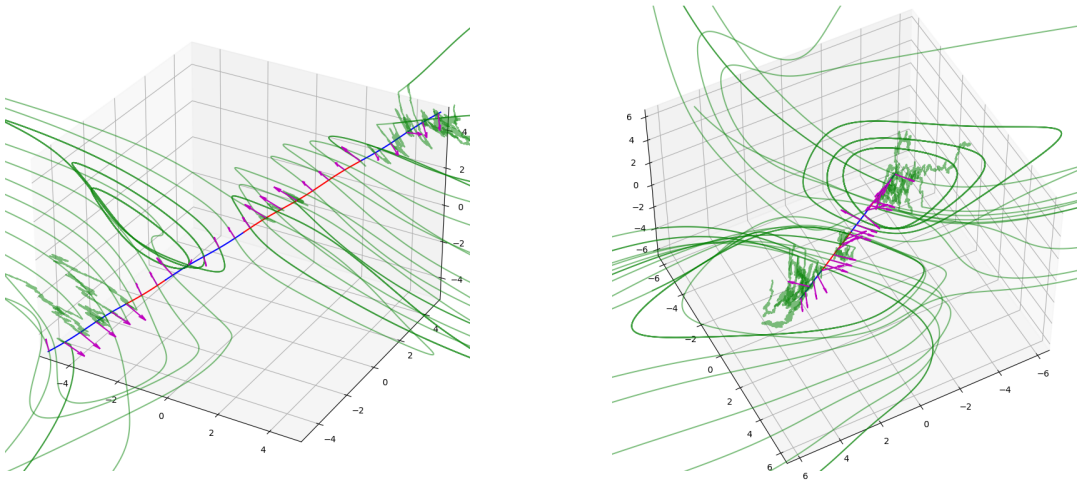


Figure 4: Kernel leaves (in green) of points belonging to the embedded state space. The magenta arrows are basis of the kernel at each of these points.

the pullback metric on the input space is

$$G_{in}(x) = J(x)^T G_{out}(N(x))J(x).$$

We assume that G_{in} has constant rank equal to 2. This is confirmed experimentally by sampling points in $\mathbb{R}^3$ and computing the rank of G_{in} at these points. We never found a point where the rank was smaller than 2. Hence, the kernel of G_{in} has constant rank equal to 1. It defines a distribution of dimension 1 over $\mathbb{R}^3$. Since a one-dimensional distribution is always integrable, this guarantees the existence of the kernel foliation. The only relevant points are the ones that are on the embedded state space or close to it. Thus, we chose several points along the embedded submanifold and compute the kernel leaf containing each point. The result is shown in Figure 4 with two points of view.

References

[1] L. Grementieri and R. Fioresi, "Model-centric Data Manifold: The Data Through the Eyes of the Model," Apr. 2021.