Canonical foliations of neural networks: application to robustness

Eliot Tron, Nicolas Couellan, Stéphane Puechmorel

Canonical foliations of neural networks: application to robustness

Eliot Tron eliot.tron@ens-lyon.fr
ENS de Lyon
Lyon, France

Nicolas Couellan nicolas.couellan@recherche.enac.fr
ENAC, Université de Toulouse,
7 Avenue Edouard Belin, 31400 Toulouse, France

Stéphane Puechmorel stephane.puechmorel@enac.fr
ENAC, Université de Toulouse,
7 Avenue Edouard Belin, 31400 Toulouse, France

Abstract

Adversarial attack is an emerging threat to the trustability of machine learning. Understanding these attacks is becoming a crucial task. We propose a new vision on neural network robustness using Riemannian geometry and foliation theory, and create a new adversarial attack by taking into account the curvature of the data space. This new adversarial attack called the *dog-leg attack* is a two-step approximation of a geodesic in the data space. The data space is treated as a (pseudo) Riemannian manifold equipped with the pullback of the Fisher Information Metric (FIM) of the neural network. In most cases, this metric is only semi-definite and its kernel becomes a central object to study. A canonical foliation is derived from this kernel. The curvature of the foliation's leaves gives the appropriate correction to get a two-step approximation of the geodesic and hence a new efficient adversarial attack. Our attack is tested on a toy example, a neural network trained to mimic the `Xor` function, and demonstrates better results that the state of the art attack presented by Zhao et al. (2019).

Keywords: Neural Networks, Robustness, Fisher Information Metric, Information geometry, Adversarial attacks.

1. Introduction

Lately there has been a growing interest in the analysis of neural network robustness and the sensitivity of such models to input perturbations (Fawzi et al. (2018); Shaham et al. (2018); Kolter and Wong (2018); Raghunathan et al. (2020)). Most of these investigations have highlighted their weakness to handle adversarial attacks (Szegedy et al. (2014)) and have proposed some means to increase their robustness. Adversarial attacks are real threats that could slow down or eventually stop the development of neural network models or their applications in contexts where robustness guarantees are needed. For example, in the specific case of aviation safety, immunization of critical systems to adversarial attacks should not only be guaranteed but also certified. Therefore, addressing the robustness of future on-board or air traffic control automated systems based on such models is a main concern.

Adversarial attacks are designed to fool classification models by introducing perturbations in the input data. These perturbations remain small and in the case of images for example, may be undetectable to the human eye. So far, most of the research effort has focused on designing such attacks in order to augment the training dataset with the constructed adversarial samples and expecting that training will be more robust. Among these methods, one can refer to the Fast Gradient Sign methods (Goodfellow et al. (2015)), robust optimization methods (Madry et al. (2019)), DeepFool (Moosavi-Dezfooli et al. (2016)), and others (Fawzi et al. (2017)). There are major drawbacks with these approaches. Most attacks are data dependent and provide no guarantees that all relevant attacks have been designed and added to the training set. Furthermore, the training procedure does not really learn the sensitivity of the model to possible attacks, it is just preventing from specific ones. However, crafting adversarial attacks by exploiting the properties of neural network learning is useful to understand the principles at play in the robustness or sensitivity of neural architecture.

Many authors consider neural network attacks and robustness properties in a flat Euclidean input space. Yet, it is commonly admitted that to learn from high dimensional data, data must lie in a low dimensional manifold (Fefferman et al. (2016)). Such manifold has its own curvature with no reason to be flat in general. Furthermore, to analyze neural network model separation capabilities and its robustness, it is critical to understand not only the topology of the decision boundaries in the input space but also the topology of iso-information regions induced by the neural network. Again, there is no reason to believe that these sub-manifolds have zero curvature in general. The Fisher information metric (FIM) is a valid metric for such purpose. Indeed, the network output is seen as a discrete probability that lies on a statistical manifold. The FIM may then be used as a Riemannian metric at the output and the pullback metric of the Fisher information as a metric for the input manifold (Zhao et al. (2019)). The importance of the FIM in the context of deep neural networks has already been pointed out by several authors. In Karakida et al. (2019), it is shown that the FIM defines the landscape of the parameter space and the maximum FIM eigenvalue defines an approximation of the appropriate learning rate for gradient methods. The FIM with respect to data (local data matrix) instead of network parameters has also been investigated from a geometric perspective in Grementieri and Fioresi (2021). The authors have shown that training data are organized on sub-manifolds (leaf) of a foliation of the data domain. A few authors have also tried to exploit this geometric knowledge to construct adversarial attacks or get some form of immunization from them. In Zhao et al. (2019), the direction of eigenvector corresponding to the maximum eigenvalue of the pullback FIM metric is used as a direction of attack where as in Shen et al. (2019), similar developments are proposed to robustify the model by regularizing the neural network loss function by the trace of the FIM.

In this work, we build on the work of (Zhao et al. (2019)) and exploit further the geometric properties of the foliation of the pullback metric of the neural network FIM. More specifically, we show that the curvature of the leaf of the foliation can be utilized to construct a two-step attack procedure referred as *dog-leg* attack. Given a budget of attack, meaning the Euclidean norm of the attack vector, we first move in the direction of the eigenvector corresponding to the maximum eigenvalue of the FIM as proposed in Zhao et al. (2019) and then make another move that takes into account the curvature. The

two steps could be seen as a discretized move along a geodesic curve. The interest of this procedure is not only to prove that, for a given budget of attack, it is possible to construct worst attacks than those proposed in Zhao et al. (2019) but also and more importantly to emphasize the role of curvature in the sensitivity of neural network models. Mathematically, this translates into the expression of the quadratic form approaching the Kullback-liebler divergence between the network output probability distribution at the origin of the attack and the probability distribution at the point reached by the attack. Indeed, we show that this expression makes explicit use of the Riemannian curvature tensor of the foliation in the input space. Experimentally, the two-step procedure is illustrated and tested against the one step attack on the `XOR` toy problem. The small experiment confirms that exploiting the curvature leads to higher fooling rates of the neural network when attacks are generated at random points with the two-step rather than the one-step strategy.

The paper is organized as follows. Section 2 details the general mathematical framework of this study and defines precisely the adversarial attacks that are considered. In the first part of Section 3, the construction of the local attack by Zhao et al. (2019) is recalled. Next, in Section 4, the main developments of this research are detailed, consisting in extending the local attack using the geometry of the problem resulting in the so-called two-step attack. Section 5 is dedicated to the application of the method to a simple play-test problem. It details more precisely the required calculations on a simple low dimensional problem and provides illustrations of the resulting foliation at the heart of the technique. Section 6 concludes the article. Appendix A gathers some of the proofs of mathematical results and Appendix B reminds some important concepts of Riemannnian geometry that are used throughout the article.

2. Problem statement

2.1 Setup

In this paper, we are studying the behavior of a neural network $N : \mathcal{X} \to \mathcal{M}$. Its output can be considered as a parameterized probability density function $p_\theta(y \mid x) \in \mathcal{M}$ where $\mathcal{M}$ is a manifold of probability density functions, $x \in \mathcal{X}$ is the input, $y \in \mathcal{Y}$ is the targeted label and $\theta \in \Theta$ is the parameter of the model (for instance the weights and biases in a perceptron). The geometric study of such probability distributions is part of *Information Geometry* (Nielsen, 2020).

In this case, $\mathcal{M}$ and $\mathcal{X}$ are two (pseudo) Riemannian Manifolds when equipped with the *Fisher Information Metric* (FIM). Fisher Information is originally a way to measure the variance of a distribution along a parameter. It was then used as a Riemannian metric by Amari, which gave birth to *Information Geometry* (see Nielsen, 2020). Measuring how the distribution of the predicted labels changes with input perturbations falls exactly in our usecase. This explains why the FIM is a good candidate to be $\mathcal{X}$ and $\mathcal{M}$'s metric.

Definition 1 (Fisher Information Metric) *The Fisher Information Metric (FIM) on the manifold $\mathcal{X}$ at the point x is defined by the following positive semi-definite symmetric matrix:*

$$(G_x)_{ij} = -\mathbb{E}_{y|x,\theta}\left[\partial_{x_i}\partial_{x_j}\left(\ln p(y \mid x, \theta)\right)\right].$$

Remark 1 *Note that this definition is not the usual definition of the Fisher Information. Indeed, the one defined by Fisher, used in the work of Amari and many other, differentiate with respect to the parameter θ where we differentiate with respect to the input x. Grementieri and Fioresi (2021) call this new metric the* local data matrix *to avoid confusion.*

An important question arising when dealing with Fisher information metric is to know when going in the converse direction is feasible. Given a Riemannian manifold $(\mathcal{X}, g)$, is it possible to find a probability family such that g is exactly its Fisher information. This is exactly what is behind the next definition.

Definition 2 (Statistical model) *A statistical model for a Riemannian manifold $(\mathcal{X}, g)$ is a probability space $(\Omega, \mathcal{T}, P)$ such that:*

- *It exists a family of probabilities p_x, $x \in \mathcal{X}$, absolutely continuous with respect to P.*
- *For any $x \in X$:*
$$g_{ij}(x) = E_{p_x}\left[\partial_i \log p_x \partial_j \log p_x\right]$$

Remark 2 *When p_x is C^2 with support not depending on x and the conditions for exchanging derivative and expectation are satisfied, then:*

$$g_{ij}(x) = -E_{p_x}\left[\partial_{ij} \log p_x\right]$$

In such a case, the metric g is Hessian.

In nearly all cases considered in machine learning, the metric g is only semi-definite. It thus makes sense to consider its kernel.

Definition 3 *Let g be a semi-definite metric on a manifold $\mathcal{X}$. A tangent vector $X \in T_x\mathcal{X}$ is said to belong to the kernel* $\ker_x g$ *of g_x if for any $Y \in T_x\mathcal{X}$, $g(X, Y) = 0$.*

Proposition 2.1 *Let $(\mathcal{X}, g)$ be a connected manifold with g a semi-definite metric. If it exists a torsionless connection ∇ on $T\mathcal{X}$ such that $\nabla g = 0$, then the mapping $x \in \mathcal{X} \to \ker_x g$ defines an integrable distribution, denoted by* $\ker g$.

Proof It is clear that for any $x \in \mathcal{X}$, $\ker_x g$ is a linear subspace of $T_x\mathcal{X}$. Let X, Y, Z be vector fields such that $Y \in \ker g$. Then, since $\nabla g = 0$ by assumption:

$$X\left(g(Y, Z)\right) = g\left(\nabla_X Y, Z\right) + g\left(Y, \nabla_X Z\right)$$

Since $Y \in \ker g$:

$$X\left(g(Y, Z)\right) = 0 = g\left(\nabla_X Y, Z\right)$$

and so, for any X, $\nabla_X Y \in \ker g$. This proves that the parallel transport of a vector in $\ker g$ is a vector in $\ker g$. The dimension of $\ker g$ is thus constant. Now, if $X, Y \in \ker g$, by the above result and since ∇ has vanishing torsion: $[X, Y] = \nabla_X Y - \nabla_Y X \in \ker g$, proving that $\ker g$ is an integrable distribution. ■

Remark 3 *If the dimension of* $\ker g$ *is not constant, then no torsionless connection* ∇ *can be such that* $\nabla g = 0$*. However, there is still a singular foliation associated with* $\ker g$*, with a canonical stratification by the dimension of* $\ker g$*.*

Proposition 2.2 *Under the assumptions of prop. 2.1,* g *defines a transverse metric for the* $\ker g$ *foliation.*

Proof This is essentially prop 3.2, p. 78 in Molino (1988). ■

The leaves of the $\ker g$ foliation are neutral submanifolds for the fisher information metric, that is moving along them will not modify the output distribution. On the other hand, the transverse metric is a measure of output variation when moving in a direction normal to the leaves. For a given euclidean budget, the efficiency of an attack will thus be higher when the transverse curvature is large.

2.2 Adversarial attacks

One primary goal of this article is to craft the best *adversarial attack* possible, or at least approach it. An adversarial attack aims at disturbing the output of the neural network by adding noise to the original input. The FIM with respect to the input is a good measure of dissimilarities between outputs, given a displacement of the input. Therefore, one would like to maximize the geodesic distance[1] $d(x_0, x_a)$ between the input point x_0 and the point x_a after the attack.

Proposition 2.3 *The geodesic distance can be expressed with the Riemannian norm and the logarithm map*[2]*:*

$$d(x_0, x_a)^2 = \left\|\log_{x_0} x_a\right\|_{\mathcal{X}}^2 = G_{x_0}\left(\log_{x_0} x_a, \log_{x_0} x_a\right).$$

The optimal solution will thus be the $\hat{x}_a$ maximizing this quantity, with some constraints. Indeed, one of the characteristics of a good adversarial attack is to be as undetectable as possible by usual measurements on the input. A usual measure for that is the Euclidean distance between x_0 and x_a. Therefore, a Euclidean constraint on a local parametrization of the attack is relevant: $\|v\|_2^2$ with $x_a = x_a(v)$ and $v \in T_{x_0}\mathcal{X}$

Proposition 2.4 *The optimal attack with a Euclidean budget of* $\varepsilon > 0$ *verifies:*

$$\max_v \left\|\log_{x_0}\left(x_a\left(v\right)\right)\right\|_{\mathcal{X}}^2 \quad \textit{subject to} \quad \|v\|_2^2 \leq \varepsilon^2.$$

A regular optimization algorithm could solve this problem, but the amount of calculus would be tremendous since the Riemannian logarithm requires to solve an ODE with boundary conditions each time. This greedy solution is thus not feasible in practice. To get close to the optimal solution, the Riemannian logarithm must be approximated.

Definition 4 (Adversarial Attack Problem)

$$\max_v \left\|\log_{x_0} x_a(v)\right\|_{\mathcal{X}}^2 \ \textit{subject to} \ \|v\|_2^2 \leq \varepsilon^2. \tag{AAP}$$

1. see Definition 10 in Appendix B
2. see Definition 13 in Appendix B

3. A local method

Zhao et al. (2019) have proposed a method to approximate the solution of a near-(AAP) problem. Their formulation of the Adversarial Attack Problem is not exactly the same though. The criteria they are maximizing is not the geodesic distance, but the Kullback-Leibler divergence. The matter with (KL) divergences is that it is not a distance, and thus not symmetric. Nevertheless, a second order Taylor approximation gives us that:

$$D_{KL}\left(p_\theta\left(y \mid x\right) \parallel p_\theta\left(y \mid x+v\right)\right) \approx \frac{1}{2} v^T G_x v = \frac{1}{2} G_x\left(v, v\right) = \frac{1}{2}\left\|v\right\|_{\mathcal{X}}^2 .$$

If we do not take into account the curvature of the input space and confound $\log_{x_0} x_a(v)$ and v, the problem solved by Zhao et al. (2019) is an approximation of (AAP). Doing so amounts to approaching geodesic by Euclidean lines in the local parametrization given by v.

Proposition 3.1 *The vector $\hat{v}$ maximizing the quadratic form $v \mapsto v^T G_x v$ is an eigenvector of the FIM G_x corresponding to the largest eigenvalue. Its re-normalization by $\frac{\varepsilon^2}{\|\hat{v}\|_2^2}$ gives an approximated solution to (AAP). This method is illustrated by Figure 1.*

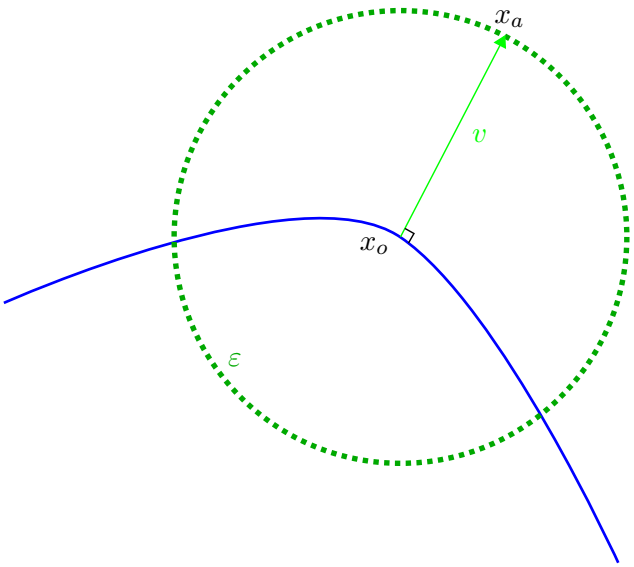


Figure 1: The one-step spectral attack in action. The circle represents the Euclidean budget. The blue curve represents the leaf of the kernel foliation.

Proof

To maximize this quadratic form under the constraint $\|v\|_2^2 = \varepsilon^2$, Karush-Kuhn-Tucker's optimality conditions[3] ensure that there exists a scalar $\lambda \in \mathbb{R}$ such that:

$$\begin{aligned} & \nabla_v \left(v^T G_x v\right) - \lambda \nabla_v \left(\|v\|_2^2 - \varepsilon^2\right) = 0 \\ \implies & \left(G_x + G_x^T\right) \hat{v} = \lambda 2 \hat{v} \\ \implies & G_x \hat{v} = \lambda \hat{v}. \end{aligned}$$

Otherwise said, the optimum $\hat{v}$ is in the set of G_x's eigenvectors. In this case:

$$\begin{aligned} \|\hat{v}\|_{\mathcal{X}}^2 &= \hat{v}^T G_x \hat{v} \\ &= \hat{v}^T \lambda \hat{v} \\ &= \lambda \|\hat{v}\|_2^2 \\ &= \lambda \varepsilon^2. \end{aligned}$$

Thus, $\hat{v}$ corresponding to the largest eigenvalue maximizes the Riemannian norm.

■

This first method is very local and does not take into account the curvature of the data. Hence, in regions of $\mathcal{X}$ where this curvature is strong, we cannot rely on this method anymore.

4. A two-step attack

The procedure detailed below is inspired by Powell's dog-leg method[4] in optimization which consists in making a first step in the gradient direction, and then a second step in the Gauss-Newton direction to correct the trajectory of the iterates. The idea here is to take a first local step v as in section 3, and then take a second step. The two-step nature of this attack allows the second step to take into account the curvature of the area around the example. This method is able to exploit the geometry of the problem to construct a better approximation of the original problem.

Let v be the solution of (AAP) using the method of section 3 but with a budget of $\mu^2 < \varepsilon^2$.

The new problem to solve is the following:

$$\max_w \|w\|_{\mathcal{X}}^2 \text{ subject to } \begin{cases} \|w + v\|_2^2 \leq \varepsilon^2 \\ \|v\|_2^2 = \mu^2 < \varepsilon^2 \\ v \text{ eigenvector of } G_x \end{cases} \tag{S2P}$$

One could solve this problem the same way as in section 3 by taking w the eigenvector of G_{x+v} with the largest eigenvalue once again (see Figure 2). By doing that, we linearly approach the geodesic between x_0 and x_a by two Euclidean lines. Taking more steps would

3. See Kuhn and Tucker (2014) for more details.
4. See Lourakis and Argyros (2005) for a detailed explanation of this method.

approximate better the geodesic. Using only two steps as proposed is a simple procedure from a computational point of view and it will achieve significantly better results than taking just one step as we will see in section 5.

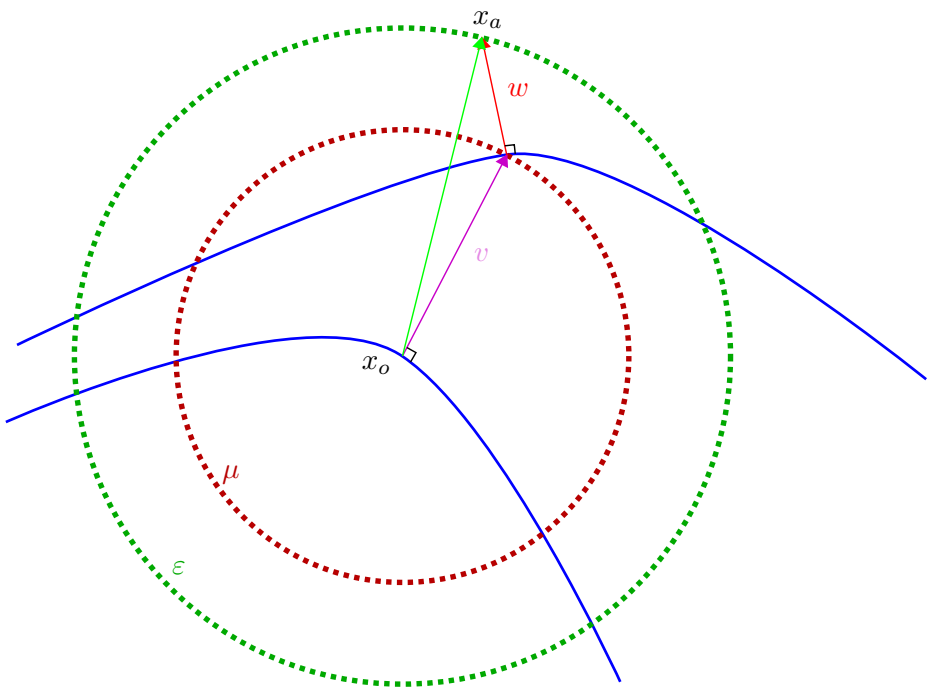


Figure 2: The dog-leg attack in action. The two circles represent the Euclidean budget. The blue curves represent the leaves of the kernel foliation.

To explicit the action of the curvature at x of the input space on the trajectory of this multi-step attack, we will approximate G_{x+v} by its value at x using normal coordinates[5]. Additionally, the calculation of G being expensive, this approximation will save computational time.

Remark 4 *In the sequel of the article, for any vector (or tensor) x, $\bar{x}$ will denote x expressed in terms of normal coordinates.*

Proposition 4.1 *If $\overline{x}$ are the normal coordinates at x_0 and if R is the Riemannian curvature tensor, then*

$$w^T G_{x+v} w \approx \frac{\partial x^m}{\partial \overline{x}^i} \frac{\partial x^n}{\partial \overline{x}^j} \left(\delta_{mn} + \frac{1}{3} \overline{R}_{mkln}(x) \overline{v}^k \overline{v}^l \right) w^i w^j \tag{1}$$

5. see Definition 12 in Appendix B

Proof

$$\begin{aligned} w^T G_{x+v} w &= (G_{x+v})_{ij}\, w^i w^j \\ &= \frac{\partial x^m}{\partial \overline{x}^i}\frac{\partial x^n}{\partial \overline{x}^j}\left(\overline{G}_{x+v}\right)_{nm} w^i w^j \\ &\approx \frac{\partial x^m}{\partial \overline{x}^i}\frac{\partial x^n}{\partial \overline{x}^j}\left(\delta_{mn} + \frac{1}{3}\overline{R}_{mkln}(x)\overline{v}^k\overline{v}^l\right) w^i w^j. \end{aligned}$$

■

Proposition 4.2 *By denoting* $\overline{w} = \left(\frac{\partial x^m}{\partial \overline{x}^i} w^i\right)_m = Pw$ *and* $R_v = \overline{R}_{mkln}(x)\overline{v}^k\overline{v}^l$*, Equation 1 can be rewritten with matrix notation by the following:*

$$\|w\|^2_{\mathcal{X}} \approx \|\overline{w}\|_2^2 + \frac{1}{3}\overline{w}^T R_v \overline{w}. \tag{2}$$

Remark 5 *In what follows,* $\|w\|^2_{\mathcal{X}}$ *will always be taken at* $x+v$*, and be approximated by the right hand of Equation 2. Additionally, we compare* w *and* v *without taking into account parallel transport since the Christoffel symbols vanish around the origin of the normal coordinates.*

The transition matrix P is equal to $\left(\frac{\partial x^i}{\partial \overline{x}^j}\right)_{i,j}$ and its inverse to $\left(\frac{\partial \overline{x}^i}{\partial x^j}\right)_{i,j}$. We should have for instance:

$$G_x = P^T \overline{G}_x P = P^T I_n P \text{ and } {P^{-1}}^T G_x P^{-1} = I_n$$

The matrix $P^{-1} = \begin{bmatrix} \frac{v_1}{\sqrt{\lambda_1}} & \cdots & \frac{v_n}{\sqrt{\lambda_n}} \end{bmatrix}$ with v_i the eigenvector of G_x associated with the eigenvalue λ_i satisfies this equation (the family is chosen to be orthonormal: $v_i^T v_j = \delta_{i,j}$). Note that this gives us

$$P = \begin{bmatrix} \sqrt{\lambda_1} v_1^T \\ \vdots \\ \sqrt{\lambda_n} v_n^T \end{bmatrix}.$$

Remark 6 *The pullback metric* G_x *is always degenerate in this problem. Indeed, the dimension of its image is strictly bounded by the number of classes of the given task*[6]*.*

To take this into account, if $d = \dim Im G_x$*, one can rewrite the metric in normal coordinates by:*

$$\overline{G}_x = \begin{bmatrix} \boldsymbol{I}_d & \boldsymbol{0}_{d,n-d} \\ \boldsymbol{0}_{n-d,d} & \boldsymbol{0}_{n-d,n-d} \end{bmatrix}.$$

The transition matrix P^{-1} *is equal to* $\begin{bmatrix} \frac{v_1}{\sqrt{\lambda_1}} & \cdots & \frac{v_d}{\sqrt{\lambda_d}} & v_{d+1} & \cdots & v_n \end{bmatrix}$.

Proposition 4.3 *If* w *is a solution to S2P, then there exists a scalar* $\lambda \geq 0$ *such that*

$$\left(P^T B P - \lambda I_n\right) w = \lambda v \tag{3}$$

with $B = I_n + \frac{1}{3}R_v$.

6. See (Grementieri and Fioresi, 2021) for the proof.

Proof Using again the KKT conditions but with the constraint that $\|v+w\|_2^2 = \left\|v+P^{-1}\overline{w}\right\|_2^2 \leq \varepsilon^2$, it implies that there exists a scalar $\lambda \geq 0$ such that:

$$\begin{aligned} &\nabla_{\overline{w}}\left(\|\overline{w}\|_2^2 + \frac{1}{3}\overline{w}^T R_v \overline{w}\right) - \lambda\nabla_{\overline{w}}\left(\left\|v+P^{-1}\overline{w}\right\|_2^2 - \varepsilon^2\right) = 0 \\ \Longrightarrow\ &\overline{w} + \frac{1}{3}R_v\overline{w} - \lambda\left(P^{-1^T}\left(P^{-1}\overline{w}+v\right)\right) = 0. \end{aligned}$$

Thus

$$\begin{aligned} &P^T\left(I_n + \frac{1}{3}R_v\right)Pw = \lambda\left(w+v\right) \\ \Longleftrightarrow\ &P^T\left(B - \lambda P^{-1^T}P^{-1}\right)Pw = \lambda v \\ \Longleftrightarrow\ &\left(P^TBP - \lambda I_n\right)w = \lambda v \end{aligned}$$

where $B = I_n + \frac{1}{3}R_v$. ■

Corollary 4.3.1 *In normal coordinates, Equation 3 rewrites*

$$PP^TB\overline{w} = \lambda\left(\overline{w}+\overline{v}\right) \ \ ie\ \ \left(PP^TB - \lambda I_n\right)\overline{w} = \lambda\overline{v}.$$

Remark 7 *Note that* $PP^T = \mathrm{diag}\left(\lambda_i\right)$.

Corollary 4.3.2 *Whenever* λ *is not an eigenvalue of* P^TBP, $\left(P^TBP - \lambda I_n\right)$ *is non-singular and*

$$w = \lambda\left(P^TBP - \lambda I_n\right)^{-1}v. \tag{4}$$

It remains to be found a λ such that the constraint $\|w+v\|_2^2 \leq \varepsilon^2$ is satisfied. There are two cases:

1. either $\|w+v\|_2^2 = \varepsilon^2$ and $\lambda > 0$,
2. or $\|w+v\|_2^2 < \varepsilon^2$ and $\lambda = 0$.

4.1 The case $\lambda > 0$

We will use the constraint $\|w+v\|_2^2 = \varepsilon^2$ to find λ. We suppose in what follows that 2λ is not an eigenvalue of B, ie $(2\lambda - 1)$ is not an eigenvalue of $\frac{1}{3}R_v$. The constraint may be written as:

$$\begin{aligned} \|w+v\|_2^2 &= \left\|\lambda\left(P^TBP - \lambda I_n\right)^{-1}v + v\right\|_2^2 \\ &= \left\|\left(\lambda\left(P^TBP - \lambda I_n\right)^{-1} + I_n\right)v\right\|_2^2. \end{aligned}$$

Lemma 4.4

$$\lambda\left(P^TBP - \lambda I_n\right)^{-1} + I_n = \left(P^TBP - \lambda I_n\right)^{-1}P^TBP.$$

Proof

$$\begin{aligned}\left(P^TBP - \lambda I_n\right)\left(\lambda\left(P^TBP - \lambda I_n\right)^{-1} + I_n\right) &= \lambda\left(P^TBP - \lambda I_n\right)\left(P^TBP - \lambda I_n\right)^{-1} + \left(P^TBP - \lambda I_n\right) \\ &= \cancel{\lambda I_n} + P^TBP - \cancel{\lambda I_n} \\ &= P^TBP.\end{aligned}$$

■

Thus,

$$\varepsilon^2 = \|w + v\|_2^2 = \left\|P^TBP\left(P^TBP - \lambda I_n\right)^{-1} v\right\|_2^2. \tag{5}$$

To find λ that satisfies Equation 5, we will study the vanishing points of:

$$\begin{array}{rccc}\varphi & : & [0,\infty[& \longrightarrow & [0,\infty[\\ & & \lambda & \longmapsto & \left\|P^TBP\left(P^TBP - \lambda I_n\right)^{-1} v\right\|_2^2 - \varepsilon^2\end{array}$$

However, finding the vanishing points of such a function is not an easy task. Several methods may be used. A numerical method such as the Newton's method (Dennis Jr and Schnabel, 1996) could be applied.

Alternatively, observe that problem S2P can be simplified by using the triangular inequality to get back to an easier eigenvalue problem. Indeed, consider the following problem:

$$\max_w \|w\|_{\mathcal{X}}^2 \text{ subject to } \begin{cases}\|w\|_2 \leq \varepsilon - \mu \\ \|v\|_2^2 = \mu^2 < \varepsilon^2 \\ v \text{ eigenvector of } G_x\end{cases} \tag{S2.2P}$$

This new problem is illustrated on Figure 3 below. The green circle is the true budget and the two other smaller circles represent the triangular inequality approximation. One can see that the second step w does not reach the green circle but stop before due to the approximation.

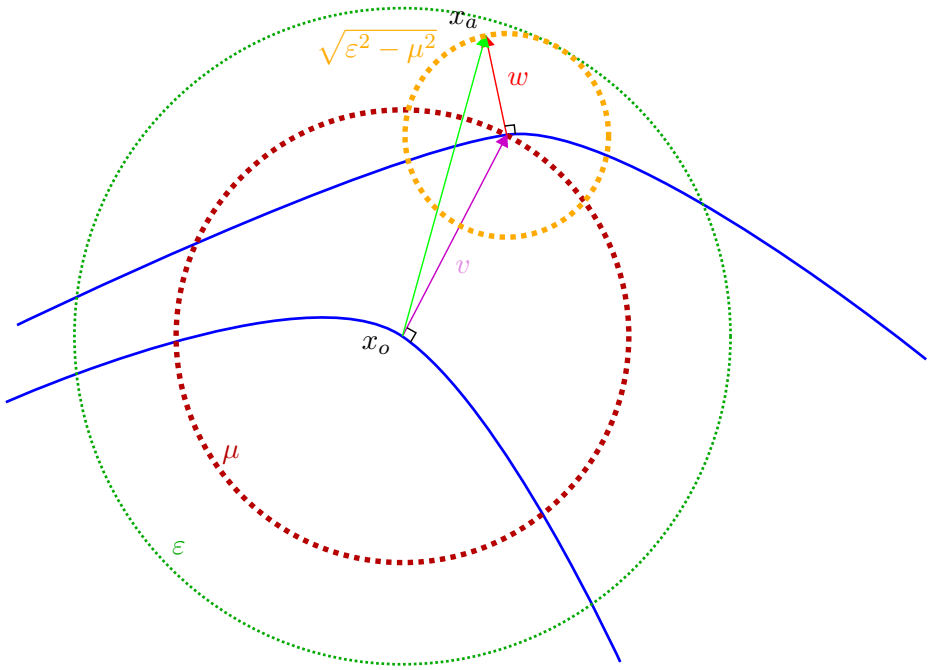


Figure 3: The dog-leg attack in action with the triangular inequality simplification. The three circles represent the Euclidean budget. The blue curves represent the leaves of the kernel foliation.

Proposition 4.5 *Any solution of S2.2P problem will satisfy the constraint of S2P problem.*

Proof Indeed, by the triangular inequality, $\|w + v\|_2 \leq \|w\|_2 + \|v\|_2 \leq (\varepsilon - \mu) + \mu = \varepsilon$. ■

With this new constraint, KKT's conditions boil down to

$$P^T B P w = \lambda w$$

or in normal coordinates:

$$P P^T B \overline{w} = \lambda \overline{w}.$$

Proposition 4.6 *A solution to S2.2P is to choose $\overline{w}$ to be the eigenvector of PP^TB with the highest eigenvalue λ and with the appropriate Euclidean norm of $\varepsilon - \mu$.*

4.2 The case $\lambda = 0$

In that case, w is in the interior of the boundary of the problem. The problem reduces to

$$B\overline{w} = 0$$
$$\iff R_v \overline{w} = -3\overline{w}.$$

This means that w is the optimal second step if $\overline{w} \in \operatorname{Ker} B$, ie if w is eigenvector of R_v with eigenvalue -3. However, this case does not produce any interesting KKT admissible point. Indeed,

$$\begin{aligned}\|w\|_{\mathcal{X}}^2 &= \|\overline{w}\|_2^2 + \frac{1}{3}\overline{w}^T R_v \overline{w} \\ &= \|\overline{w}\|_2^2 - \overline{w}^T\overline{w} \\ &= 0.\end{aligned}$$

Therefore, the case $\lambda = 0$ does not lead to useful adversarial attacks when the previous approximations are applied to the problem. The study of this singularity is left as future work.

5. A simple play-test example

5.1 Setup

In this section, we compare numerically the efficiency of the local method (section 3), the two-step method (section 4) when the FIM is recomputed at the intermediary point and the two-step method when the normal coordinates approximation is used (Equation 1 of section 4). These three methods are tested against a simple neural network N_θ with one hidden layer of k neurons and sigmoids as activation functions. Its architecture is depicted on Figure 4.

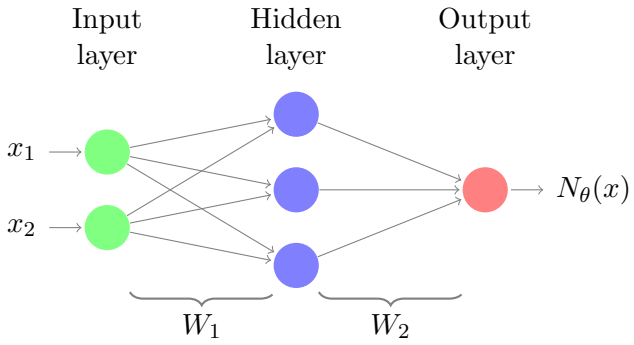


Figure 4: XorNet N_θ with 3 hidden neurons.

To be more precise, $\sigma : a \in \mathbb{R}^d \mapsto \left(\frac{1}{1+e^{-a_i}}\right)_{i=1}^d \in \mathbb{R}^d$ is the sigmoid function and if $x \in \mathbb{R}^2$, $W_1 \in \mathcal{M}_{2,k}(\mathbb{R})$, $W_2 \in \mathcal{M}_{k,1}(\mathbb{R})$, we have

$$N_\theta(x) = \sigma\left(W_2\sigma\left(W_1 x + b_1\right) + b_2\right). \tag{6}$$

This neural network is then trained to approximate the very simple function `Xor`: $\{0,1\} \to \{0,1\}$ defined in Table 1.

`Xor`	0	1
0	0	1
1	1	0

Table 1: `Xor` function.

The output $N_\theta(x)$ is seen as the parameter p of a Bernoulli law and associates to the network the following probability distribution: $p(y \mid x, \theta)$ where θ is the vector containing the weights and biases $(W_i,\ b_i)$, x is the input and y is the true label.

Proposition 5.1 *The random variable $y \mid x, \theta$ follows Bernoulli's law of parameter $p = N_\theta(x) = p\,(1 \mid x, \theta)$.*

5.2 Computing the output FIM

The output of the network is the manifold of Bernoulli probability densities parameterized by the open segment $]0, 1[$.

Proposition 5.2 *Let $p \in]0, 1[$ and G_p the Fisher Information Metric at the point p.*

$$G_p = \frac{1}{p}\frac{1}{1-p} \tag{7}$$

Proof

$$\begin{aligned}
G_p &= -\mathbb{E}_{y|x,\theta}\left[\partial_p^2\left(\ln p(y \mid x, \theta)\right)\right] \\
&= -\mathbb{E}\left[\partial_p^2\left(y \ln p + (1-y)\ln(1-p)\right)\right] \\
&= -\mathbb{E}\left[\partial_p\left(\frac{y}{p} - \frac{(1-y)}{1-p}\right)\right] \\
&= -\mathbb{E}\left[-\frac{y}{p^2} - \frac{(1-y)}{(1-p)^2}\right] \\
&= \frac{1}{p} + \frac{1}{1-p} \\
&= \frac{1}{p}\frac{1}{1-p}.
\end{aligned}$$

■

5.3 Computing the pullback metric

Let $x \in \mathcal{M}$ be a point associated with p by the network.

Lemma 5.3 *The Fisher Information Metric G_x on $\mathcal{X}$ is the pullback metric of G_p by the neural network N_θ.*

Corollary 5.3.1 *If $J = \left[\frac{\partial p}{\partial x_j}\right]_{j=1,2} = \left[\frac{\partial N_\theta(x)}{\partial x_1} \quad \frac{\partial N_\theta(x)}{\partial x_2}\right]$, then*

$$G_x = J^T G_p J. \tag{8}$$

Proposition 5.4 *If $a = \sigma'\left(W_2\sigma\left(W_1 x + b_1\right) + b_2\right)$ and $\Sigma = \operatorname{diag}\left(\sigma'\left(W_1 x + b_1\right)_j,\ 1 \le j \le k\right)$, then*

$$J = aW_2\Sigma W_1 \quad \textit{and} \quad G_x = \frac{1}{p}\frac{1}{1-p}J^T J. \tag{9}$$

Proof

$$\begin{aligned}
\frac{\partial N_\theta(x)}{\partial x_j} &= \sigma'\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)\frac{\partial W_2\sigma\left(W_1x+b_1\right)+b_2}{\partial x_j} \\
&= \sigma'\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)\sum_{i=1}^{k}\left(W_2\right)_i\frac{\partial\sigma\left(W_1x+b_1\right)_i}{\partial x_j} \\
&= \sigma'\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)\sum_{i=1}^{k}\left(W_2\right)_i\sigma'\left(W_1x+b_1\right)_i\frac{\partial\left(W_1x+b_1\right)_i}{\partial x_j} \\
&= \sigma'\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)\sum_{i=1}^{k}\left(W_2\right)_i\sigma'\left(W_1x+b_1\right)_i\left(W_1\right)_{i,j} \\
&= aW_2\Sigma\left(W_1\right)_{\cdot,j}
\end{aligned}$$

where $a=\sigma'\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)$ and $\Sigma=\operatorname{diag}\left(\sigma'\left(W_1x+b_1\right)_j,\ 1\le j\le k\right)$.

■

Knowing how to compute the metric at any point unlocks the computation of the local method seen in section 3 and the two-step method seen in section 4 when the FIM is recomputed at the intermediary point. To use the information at the origin of the attack, it is necessary to compute the curvature on the data space in normal coordinates as seen in Equation 1 of section 4. Additionally, studying the curvature of the input space is insightful to understand the behavior of the correction step in the two-step attack, and more generally to understand why adversarial attacks are, in some cases, so efficient.

Nonetheless, the pullback metric being almost always only semi-definite for machine learning tasks, it makes sense to consider its kernel. The curvature will have a decomposition term on the kernel of g and a decomposition term on its orthogonal. To craft the attack, the orthogonal term will be the only curvature component of interest as stated at the end of Section 2.1. Thus in the following subsection, we compute the metric kernel for this neural network.

5.4 The metric kernel foliation

Definition 5 *The kernel of a metric G at the point x is defined by*

$$\ker_x G=\left\{X\in\mathcal{T}_p\mathcal{M}\mid X^TG_xY=0,\ \forall Y\in\mathcal{T}_p\mathcal{M}\right\}.$$

This kernel defines an integrable distribution when Frœbenius' condition[7] is satisfied, and with this distribution emerges a Riemannian foliation on the input manifold, foliation defined by the action of the neural network.

Lemma 5.5 $\ker_x G=\ker J$.

7. For more details, see Chapter 1 of Molino

Proof (of the lemma). We omit $\frac{1}{p}\frac{1}{1-p}$ during the proof because it is always non-zero.

- Let us prove first that $\ker_x G \subset \ker J$. If $X \in \ker_x G$, then $X^T J^T J X = 0$. Hence $(JX)^T (JX) = 0$, or written otherwise: $\|JX\| = 0$. Hence $JX = 0$ and $X \in \ker J$.
- Then we can prove that $\ker J \subset \ker_x G$. This inclusion is simply due to the fact that if $X \in \ker J$, we have for all $Y \in \mathcal{T}_p\mathcal{M}$ that $X^T J^T J Y = \underbrace{(JX)^T}_{=0} JX = 0$.

■

Let $U \in \ker J$. Remind that

$$a = \sigma'\left(W_2\sigma\left(W_1 x + b_1\right) + b_2\right) \text{ and } \Sigma = \operatorname{diag}\left(\sigma'\left(W_1 x + b_1\right)_j,\ 1 \le j \le k\right).$$

Then

$$\begin{aligned} & JU = 0 \\ \iff\ & aW_2\Sigma W_1 U = 0 \\ \iff\ & W_2\Sigma W_1 U = 0. \qquad \text{(because } a \neq 0\text{)} \end{aligned}$$

Writing $W_2\Sigma W_1 = \begin{bmatrix}\alpha_1(x) & \alpha_2(x)\end{bmatrix}$ and $U = \begin{bmatrix}u_1 \\ u_2\end{bmatrix}$, one gets the following equality:

$$\alpha_1(x)u_1 + \alpha_2(x)u_2 = 0.$$

Proposition 5.6 *If at least one of the $\alpha_j(x)$ is non-zero, the distribution at x is one dimensional and given by $P_x = \operatorname{Span}\left(\alpha_2(x)\partial_1 - \alpha_2(x)\partial_1\right)$.*

Remark 8 *If $\alpha_1(x) = \alpha_2(x) = 0$, the leaf at x is singular and is of dimension 0.*

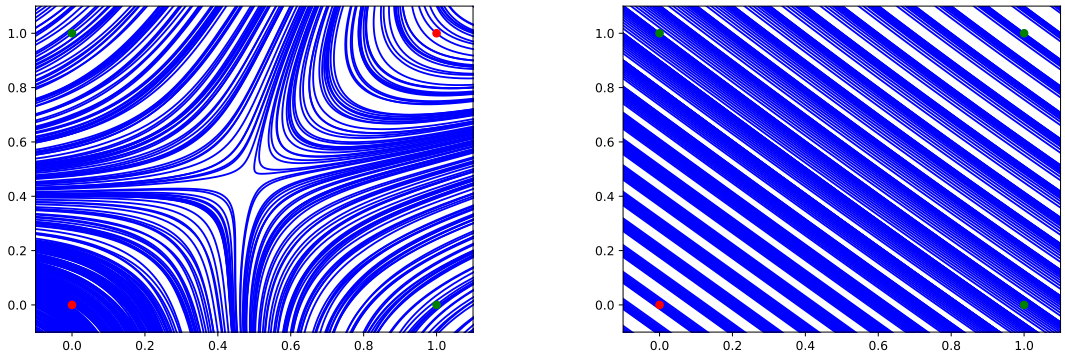
With the `Xor` function, the dimension of the leaves is 1, thus the condition of Frœbenius is trivially satisfied and P is integrable.

Proposition 5.7 *If $\gamma : t \in I \mapsto \gamma(t) \in \mathcal{M}$ is an integral curve for the distribution P, it satisfies the following PDE:*

$$\begin{aligned} \gamma'(t) &= \alpha_2\left(\gamma(t)\right)\partial_1 - \alpha_1\left(\gamma(t)\right)\partial_2 \\ &= W_2\Sigma_{\gamma(t)}W_1\begin{bmatrix}0 & -1 \\ 1 & 0\end{bmatrix}. \end{aligned}$$

It can be solved numerically quite easily with a finite difference method[8]. Figure 5 provides illustrations of the computed neural network kernel foliation with such a method for the `Xor` function (Figure 5a), and for the `Or` function (Figure 5b).

8. See for example Demailly (2016).



(a) Task: `Xor` (b) Task: `or`

Figure 5: Kernel foliation: the leaves are represented by the blue lines, the red dots are the 0 result and the green dots are the 1 results.

Remark 9 *For readability, the transverse leaves are not represented but can be easily deduced from this figures.*

These results are quite interesting. First of all, we notice that a linearly separable problem such as the `or` function seems to have hyperplanes as leaves. In fact, it is easy to show that if the neural network is replaced by a linear form $x \mapsto \langle n, x \rangle + b$, then the leaves are hyperplanes defined by the normal vector n.

Second of all, one can see that there is a singular point close to the middle point $(0.5, 0.5)$ for the `Xor` task. Clearly, it is also easy to see that around that central point, the curvature of the leaves is the highest. Since the two-step adversarial attack makes use of the curvature, we can conjecture that it is in the central region that the second step will have the most impact. This phenomena will be confirmed in subsection 5.6.

5.5 Computing the curvature

In this section, we are computing the extrinsic curvature of a leaf in order to compute the two-step attack. But first, we compute the Levi-Civita connection. In the following of the article, we will use $i, j, k, l \in \{1, \dots, \dim \mathcal{X}\}$ as indices and we will use the Einstein summation notation.

Definition 6 (Levi-Civita connection) *In coordinates e_i, if Γ_{ij}^k are Chrystoffel's symbols, then the Levi-Civita connection is defined by:*

$$\mathring{\nabla}_{e_j} e_i = \Gamma_{ij}^k e_k \quad .$$

Proposition 5.8 *When the activation function σ is the sigmoïd function, the Christoffel symbols of the Levi-Civita connection for the FIM at point $x \in \mathcal{X}$ are:*

$$\Gamma_{ij}^k = \Bigg[\left((W_2\sigma(W_1x + b_1) + b_2) - p - \frac{1}{2} \right) \left(\alpha_i(x)\delta_j^k + \alpha_j(x)\delta_i^k - \alpha^k(x)g_{ij} \right) + p(1-p)\frac{\partial \alpha_i(x)}{\partial x_j}\alpha^k(x) \Bigg]$$

Definition 7 (Riemannian curvature) *The* Riemannian Curvature Tensor *R is defined such that for all three tensor fields X, Y, Z,*

$$R(X,Y)Z = \nabla_X \nabla_Y Z - \nabla_Y \nabla_X Z - \nabla_{[X,Y]} Z.$$

In local coordinates, this gives:

$$\begin{aligned} R_{ijk}^l &= dx^l \left(R\left(\partial_i, \partial_j\right) \partial_k \right) \\ &= dx^l \left(\nabla_{\partial_i} \nabla_{\partial_j} \partial_k - \nabla_{\partial_j} \nabla_{\partial_i} \partial_k - \nabla_{[\partial_i, \partial_j]} \partial_k \right) \\ &= \frac{\partial \Gamma_{kj}^l}{\partial x^i} - \frac{\partial \Gamma_{ki}^l}{\partial x^j} + \Gamma_{kj}^\alpha \Gamma_{\alpha i}^l - \Gamma_{ki}^\alpha \Gamma_{\alpha j}^l. \end{aligned}$$

Remark 10 *In normal coordinates, at the origin point, one has:*

$$R_{ijkl} = \frac{1}{2} \left(\partial_i \partial_l g_{jk} + \partial_j \partial_k g_{il} - \partial_i \partial_k g_{jl} - \partial_j \partial_l g_{ik} \right).$$

Finding an explicit form for R is unreasonable, but since we are in the simpler case where the dimension of the transverse leaves is 1, a quick approximation can do the trick. In fact, leaves are embedded submanifolds of the euclidean space and the extrinsic curvature, i.e. the second fundamental form, can be computed as the rate of rotation of the normal vector.

To approximate the second step w of the two-step attack, we look at the rotation speed of the unit normal to the kernel leaf $\vec{n}$ when moved by an infinitesimal step dx. This infinitesimal step is taken in the direction of the transverse leaf, and is Euclidean. Since the rotation rate is approximated by finite difference, one can expect the angle variation to be very small. To ensure numerical stability, the usual procedure based on its cosine computation using inner product is replaced by one using the cross product. The next lemma gives the expression of the sine of the angle variation between two close positions on curve transverse to the $\ker g$ foliation.

Lemma 5.9 *If $\vec{n}_y$ is the normal to the kernel leaf at $y \in \mathcal{X}$, and if $\cdot \times \cdot$ is the cross product, then the infinitesimal variation of angle is*

$$|d\theta| = \arcsin\left(\|\vec{n}_x \times \vec{n}_{x+dx}\|_2 \right).$$

Please note that in the small angles approximation, the sine can be replaced by the angle itself, thus recovering the usual infinitesimal rotation representation as a cross product (this is in fact a Lie algebra representation in the usual sense).

To approximate the effect of the curvature during the Euclidean step v, one has to compute the rotation matrix R of angle $d\theta$:

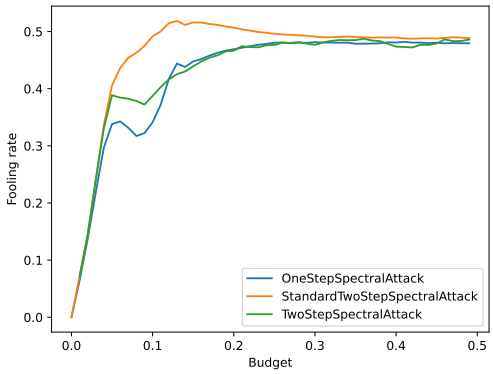
$$R = \begin{bmatrix} \cos(d\theta) & -\sin(d\theta) \\ \sin(d\theta) & \cos(d\theta) \end{bmatrix}$$

Proposition 5.10 *If v is the first step, the approximated second step is given by $w = Rv$ ands then re-normalized to get $\|w\|^2 + \|v\|^2 = \varepsilon^2$.*

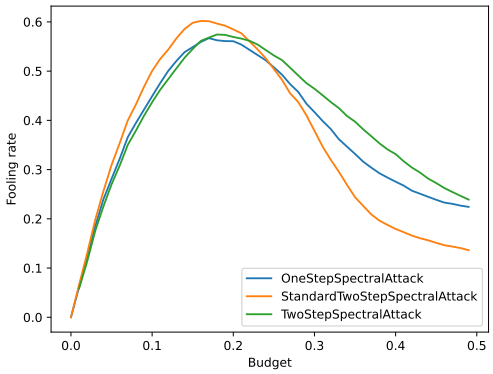
The attack $v + w$ is what we will call the *TwoStepSpectralAttack* in section 5.6.

5.6 Numerical results

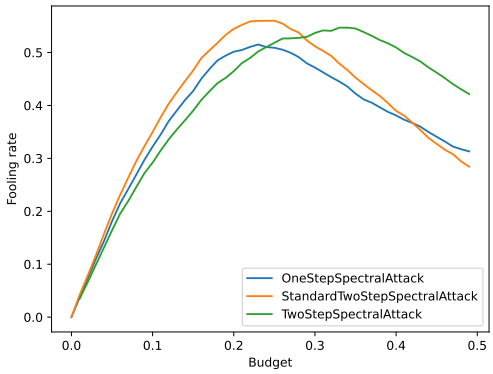
To compare the different attacks, we train a neural network with 150 hidden neurons on 5000 random points taken in the square $[0, 1]^2$. We then compute the three different attacks which are the One Step Spectral Attack presented in section 3, the Standard Two Step Spectral Attack presented at the beginning of section 4 where we compute the FIM at x_o and $x_o + v$, and the Two Step Spectral Attack where we approximate the curvature as explained in subsection 5.5. These attacks are computed on 5000 random points selected in a square I of variable length. The fooling rate is then computed as the quotient of the number of fooled prediction by the total number of points. The budget is selected between 0 and 0.5. The first step in the two-step attacks is set to have Euclidean budget of 80% of the entire budget. We plot the results below on Figure 6.



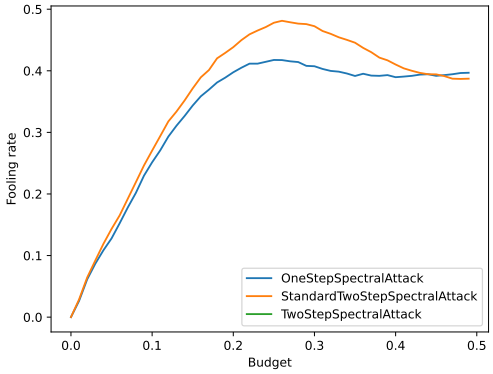
(a) $I = [0.45, 0.55]^2$



(b) $I = [0.3, 0.7]^2$



(c) $I = [0.15, 0.85]^2$



(d) $I = [0, 1]^2$

Figure 6: Fooling rates with respect to the Euclidean budget with random points taken in I.

On each figure, the two-step attacks tend to beat the one step attack, at least for the smallest budgets. The two-step attack is especially strong on the area where the curvature is the strongest (see Figure 6a). Indeed, as seen on Figure 5a, the curvature is really strong at the center of the square $[0, 1]^2$. The eigenvector of the FIM associated with the greatest eigenvalue is always orthogonal to the leaf kernel. Therefore, striking close to the middle region without taking into account the curvature usually results in not changing the output label: $(0, 0) \leftrightarrow (1, 1)$ or $(1, 0) \leftrightarrow (0, 1)$. This is why the two-step attacks get better results. The fact that the Standard Two Step Spectral Attack and the approximated one get similar results, with the first one always a bit better is not surprising since the second one is a close approximation of the first one.

These plots confirm our intuition that using the information of local curvature to craft an adversarial attack is meaningful. It also also highlights the role of the leaves of the kernel foliation in the sensitivity of neural networks to attacks.

6. Conclusion

This paper explores the relationship between adversarial attacks and curvature of the data space. Using the curvature information, we have proposed a two-steps attack that achieves better results than the One Step Spectral Attack presented by Zhao et al. (2019). The analytical mathematical expression of the proposed attack explicitly uses the curvature tensor of the FIM kernel leaves. This emphasizes the importance of geometry in the construction of an efficient attack. Additionally, with simple experiments on a toy example, we have illustrated and confirmed that exploiting such geometrical information is relevant and actually outperforms a state of the art strategy. The mathematical construction of the proposed method opens also new opportunities for future research. Indeed, it is clear that, in the case of neural networks, the geometrical properties of the leaves of the kernel foliation is related to its robustness as explained above but more generally to its power to separate data points. Therefore, the geometry of the foliation is directly linked to the complexity of the model (i.e. the neural network architecture). A deeper analysis of these aspects should help us in gaining knowledge and some explainability of the underlying principles at play in neural network learning and more generally deep learning methods. This will be the focus of our future research.

Acknowledgments

Our work has benefited from the AI Interdisciplinary Institute ANITI. ANITI is funded by the French "Investing for the Future - PIA3" program under the Grant agreement # ANR-19-PI3A-0004.

References

Jean-Pierre Demailly. *Analyse numérique et équations différentielles-4ème Ed.* EDP sciences, 2016.

John E Dennis Jr and Robert B Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. SIAM, 1996.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. The robustness of deep networks: A geometrical perspective. *IEEE Signal Processing Magazine*, 34(6): 50–62, 2017. doi: 10.1109/MSP.2017.2740965.

Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of classifiers' robustness to adversarial perturbations. *Machine Learning*, 107(3):481–508, 2018.

Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *J. Amer. Math. Soc.*, 29:983 – 1049, 2016.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6572`.

Luca Grementieri and Rita Fioresi. Model-centric Data Manifold: The Data Through the Eyes of the Model, 2021. URL http://arxiv.org/abs/2104.13289.

Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. Universal statistics of fisher information in deep neural networks: Mean field approach. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 1032–1041. PMLR, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/karakida19a.html.

J. Zico Kolter and Eric Wong. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

Harold W Kuhn and Albert W Tucker. Nonlinear programming. In *Traces and emergence of nonlinear programming*, pages 247–258. Springer, 2014.

M.L.A. Lourakis and A.A. Argyros. Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment? In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1526–1531 Vol. 2, 2005. doi: 10.1109/ICCV.2005.128.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019. URL https://arxiv.org/abs/1706.06083.

P. Molino. *Riemannian Foliations*. Progress in Mathematics. Birkhäuser Boston, 1988. ISBN 9783764333706.

Pierre Molino. *Riemannian Foliations*. Birkhäuser Boston. ISBN 978-1-4684-8672-8 978-1-4684-8670-4. doi: 10.1007/978-1-4684-8670-4. URL http://link.springer.com/10.1007/978-1-4684-8670-4.

S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2574–2582, Los Alamitos, CA, USA, jun 2016. IEEE Computer Society. doi: 10.1109/CVPR.2016.282. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.282.

Frank Nielsen. An elementary introduction to information geometry. *Entropy*, 22(10):1100, Sep 2020. ISSN 1099-4300. doi: 10.3390/e22101100. URL http://dx.doi.org/10.3390/e22101100.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified Defenses against Adversarial Examples, 2020. URL http://arxiv.org/abs/1801.09344.

Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307:195 – 204, 2018.

Chaomin Shen, Yaxin Peng, Guixu Zhang, and Jinsong Fan. Defending Against Adversarial Attacks by Suppressing the Largest Eigenvalue of Fisher Information Matrix, 2019. URL http://arxiv.org/abs/1909.06137.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, and I. Goodfellow et al. Intriguing properties of neural networks. *International Conference on learning representations (ICLR)*, 2014.

Chenxiao Zhao, P. Thomas Fletcher, Mixue Yu, Yaxin Peng, Guixu Zhang, and Chaomin Shen. The Adversarial Attack and Detection under the Fisher Information Metric. 33: 5869–5876, 2019. ISSN 2374-3468, 2159-5399. doi: 10.1609/aaai.v33i01.33015869. URL https://aimagazine.org/ojs/index.php/AAAI/article/view/4536.

A. Proofs

A.1 Proof of Proposition 5.8

Lemma A.1 *Let $i, j, k \in \{1, \dots, \dim \mathcal{X}\}$. Then*

$$\Gamma_{ij}^k = \frac{1}{2} g^{kl} \left(\frac{\partial g_{jl}}{\partial x_i} + \frac{\partial g_{il}}{\partial x_j} - \frac{\partial g_{ij}}{\partial x_l} \right).$$

With all the previous notations, we have:

$$\begin{aligned} g_{ij} &= \frac{1}{p(1-p)} \left(J^T J \right)_{ij} \\ &= \frac{1}{p(1-p)} \left(a(x) \right)^2 \alpha_i(x) \alpha_j(x). \end{aligned}$$

Let us compute the partial derivatives of the various terms of the last equation.

$$\begin{aligned} \frac{\partial}{\partial x_l} \left(\frac{1}{p(1-p)} \right) &= -\frac{1}{p^2(1-p)^2} \frac{\partial p(1-p)}{\partial x_l} \\ &= -\frac{1}{p^2(1-p)^2} \frac{\partial N_\theta(x)(1 - N_\theta(x))}{\partial x_l} \\ &= -\frac{1}{p^2(1-p)^2} \left(J_l \left(1 - p \right) - p J_l \right) \\ &= -\frac{1}{p^2(1-p)^2} \left(J_l \left(1 - 2p \right) \right). \end{aligned}$$

$$\begin{aligned} \frac{\partial \left(a(x) \right)^2}{\partial x_l} &= 2 \frac{\partial a(x)}{\partial x_l} a(x) \\ &= 2 \frac{\partial \sigma' \left(W_2 \sigma \left(W_1 x + b_1 \right) + b_2 \right)}{\partial x_l} a(x) \\ &= 2 \sigma'' \left(W_2 \sigma \left(W_1 x + b_1 \right) + b_2 \right) \frac{\partial \left(W_2 \sigma \left(W_1 x + b_1 \right) + b_2 \right)}{\partial x_l} \sigma' \left(W_2 \sigma \left(W_1 x + b_1 \right) + b_2 \right) \\ &= 2 \sigma'' \left(W_2 \sigma \left(W_1 x + b_1 \right) + b_2 \right) \underbrace{\frac{\partial \sigma \left(W_2 \sigma \left(W_1 x + b_1 \right) + b_2 \right)}{\partial x_l}}_{= J_l} \\ &= 2 \sigma'' \left(W_2 \sigma \left(W_1 x + b_1 \right) + b_2 \right) J_l. \end{aligned}$$

$$\begin{aligned} \frac{\partial \alpha_j(x)}{\partial x_l} &= \sum_{i=1}^{k} \left(W_2 \right)_i \frac{\partial \sigma' \left(W_1 x + b_1 \right)_i}{\partial x_l} \left(W_1 \right)_{ij} \\ &= \sum_{i=1}^{k} \left(W_2 \right)_i \sigma'' \left(W_1 x + b_1 \right)_i \left(W_1 \right)_{il} \left(W_1 \right)_{ij} \\ &= W_2 \Sigma' \left(\left(W_1 \right)_{\cdot, l} \odot \left(W_1 \right)_{\cdot, j} \right) \end{aligned}$$

with $\odot$ the component-wise product (Hadamard product).

Lemma A.2 *Let $i, j \in \{1, \dots, \dim \mathcal{X}\}$, then $\frac{\partial \alpha_j(x)}{\partial x_l} = \frac{\partial \alpha_l(x)}{\partial x_j}$ and the associated matrix is thus symmetric.*

Finally, we get:

$$\begin{aligned}
\frac{\partial g_{ij}}{\partial x_l} &= \frac{\partial}{\partial x_l}\left(\frac{1}{p(1-p)}\right)(a(x))^2\,\alpha_i(x)\alpha_j(x) \\
&\quad + \frac{1}{p(1-p)}\frac{\partial\,(a(x))^2}{\partial x_l}\alpha_i(x)\alpha_j(x) \\
&\quad + \frac{1}{p(1-p)}(a(x))^2\frac{\partial \alpha_i(x)}{\partial x_l}\alpha_j(x) \\
&\quad + \frac{1}{p(1-p)}(a(x))^2\alpha_i(x)\frac{\partial \alpha_j(x)}{\partial x_l} \\
\frac{\partial g_{ij}}{\partial x_l} &= -\frac{1}{p^2(1-p)^2}J_l\,(1-2p)\,(a(x))^2\,\alpha_i(x)\alpha_j(x) \\
&\quad + \frac{1}{p(1-p)}2\sigma''\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)J_l\alpha_i(x)\alpha_j(x) \\
&\quad + \frac{1}{p(1-p)}(a(x))^2\frac{\partial \alpha_i(x)}{\partial x_l}\alpha_j(x) \\
&\quad + \frac{1}{p(1-p)}(a(x))^2\alpha_i(x)\frac{\partial \alpha_j(x)}{\partial x_l}.
\end{aligned}$$

This gives:

$$\begin{aligned}
\frac{\partial g_{ij}}{\partial x_l} = \frac{1}{p(1-p)}\Bigg[&- J_l\,(1-2p)g_{ij} \\
&+ 2\sigma''\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)J_l\alpha_i(x)\alpha_j(x) \\
&+ (a(x))^2\left(\frac{\partial \alpha_i(x)}{\partial x_l}\alpha_j(x) + \alpha_i(x)\frac{\partial \alpha_j(x)}{\partial x_l}\right)\Bigg].
\end{aligned}$$

Remark 11 *In the case where σ is the sigmoid, then*

$$\sigma''(x) = \sigma'(x)\left(1-2\sigma(x)\right) = \sigma(x)\left(1-\sigma(x)\right)\left(1-2\sigma(x)\right).$$

With this remark in mind, we can simplify the computations:

$$\begin{aligned}
2\sigma''\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)J_l\alpha_i(x) &= 2\left(\sigma'\cdot(1-2\sigma)\right)\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)J_l\alpha_i(x)\alpha_j(x) \\
&= 2a(x)\left(\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)-2N_\theta(x)\right)J_l\alpha_i(x)\alpha_j(x) \\
&= 2\left(\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)-2p\right)a(x)a(x)\alpha_l(x)\alpha_i(x)\alpha_j(x) \\
&= 2\left(\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)-2p\right)\alpha_l(x)p\,(1-p)\,g_{ij}.
\end{aligned}$$

$$\frac{\partial g_{ij}}{\partial x_l} = \frac{1}{p(1-p)}\Bigg[- J_l\,(1-2p)g_{ij} \\ + 2\left((W_2\sigma\left(W_1x+b_1\right)+b_2)-2p\right)\alpha_l(x)p\,(1-p)\,g_{ij} \\ + (a(x))^2\left(\frac{\partial\alpha_i(x)}{\partial x_l}\alpha_j(x)+\alpha_i(x)\frac{\partial\alpha_j(x)}{\partial x_l}\right)\Bigg].$$

$$\Gamma^k_{ij} = \frac{1}{2p(1-p)}g^{kl}\Bigg[- J_i\,(1-2p)g_{jl} \\ + 2\left((W_2\sigma\left(W_1x+b_1\right)+b_2)-2p\right)\alpha_i(x)p\,(1-p)\,g_{jl} \\ + (a(x))^2\left(\frac{\partial\alpha_j(x)}{\partial x_i}\alpha_l(x)+\alpha_j(x)\frac{\partial\alpha_l(x)}{\partial x_i}\right) \\ - J_j\,(1-2p)g_{il} \\ + 2\left((W_2\sigma\left(W_1x+b_1\right)+b_2)-2p\right)\alpha_j(x)p\,(1-p)\,g_{il} \\ + (a(x))^2\left(\frac{\partial\alpha_i(x)}{\partial x_j}\alpha_l(x)+\alpha_i(x)\frac{\partial\alpha_l(x)}{\partial x_j}\right) \\ + J_l\,(1-2p)g_{ij} \\ - 2\left((W_2\sigma\left(W_1x+b_1\right)+b_2)-2p\right)\alpha_l(x)p\,(1-p)\,g_{ij} \\ - (a(x))^2\left(\frac{\partial\alpha_i(x)}{\partial x_l}\alpha_j(x)+\alpha_i(x)\frac{\partial\alpha_j(x)}{\partial x_l}\right)\Bigg]$$

Which can be factorized into

$$\Gamma^k_{ij} = \frac{1}{2p(1-p)}g^{kl}\Bigg[- (1-2p)\left(J_ig_{jl}+J_jg_{il}-J_lg_{ij}\right) \\ + 2\left((W_2\sigma\left(W_1x+b_1\right)+b_2)-2p\right)p\,(1-p)\left(\alpha_i(x)g_{jl}+\alpha_j(x)g_{jl}-\alpha_lg_{ij}\right) \\ + (a(x))^2\Bigg[\alpha_l(x)\left(\frac{\partial\alpha_j(x)}{\partial x_i}+\frac{\partial\alpha_i(x)}{\partial x_j}\right) \\ + \alpha_j(x)\left(\frac{\partial\alpha_l(x)}{\partial x_i}-\frac{\partial\alpha_i(x)}{\partial x_l}\right) \\ + \alpha_i(x)\left(\frac{\partial\alpha_l(x)}{\partial x_j}-\frac{\partial\alpha_j(x)}{\partial x_l}\right)\Bigg]\Bigg]$$

We can simplify the last line thanks to the remark we have made before: $\frac{\partial\alpha_j(x)}{\partial x_l} = \frac{\partial\alpha_l(x)}{\partial x_j}$, thus giving us:

$$\Gamma^k_{ij} = \frac{1}{2p(1-p)}g^{kl}\Bigg[- (1-2p)\left(J_ig_{jl}+J_jg_{il}-J_lg_{ij}\right) \\ + 2\left((W_2\sigma\left(W_1x+b_1\right)+b_2)-2p\right)p\,(1-p)\left(\alpha_i(x)g_{jl}+\alpha_j(x)g_{il}-\alpha_l(x)g_{ij}\right) \\ + 2(a(x))^2\frac{\partial\alpha_i(x)}{\partial x_j}\alpha_l(x)\Bigg]$$

Hence

$$\Gamma_{ij}^k = \frac{1}{2p(1-p)}\Bigg[-(1-2p)\left(J_i\delta_j^k + J_j\delta_i^k - J^k g_{ij}\right) \\ + 2\left(\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)-2p\right)p\left(1-p\right)\left(\alpha_i(x)\delta_j^k + \alpha_j(x)\delta_i^k - \alpha^k(x)g_{ij}\right) \\ + 2(a(x))^2\frac{\partial\alpha_i(x)}{\partial x_j}\alpha^k(x)\Bigg]$$

We can simplify considering the fact that $J = a(x)\alpha(x)$:

$$\Gamma_{ij}^k = \frac{1}{2p(1-p)}\Bigg[\left(2\left(\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)-2p\right)p\left(1-p\right) - a(x)(1-2p)\right)\left(\alpha_i\delta_j^k + \alpha_j\delta_i^k - \alpha^k g_{ij}\right) \\ + 2(a(x))^2\frac{\partial\alpha_i(x)}{\partial x_j}\alpha^k(x)\Bigg]$$

In the following, we will write A_{ij}^k for what is inside the big brackets to simplify the reading.

Remark 12 *We notice that* $a(x) = \sigma'\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right) = N_\theta(x)\left(1-N_\theta(x)\right) = p(1-p)$ *if* σ *is the sigmoid.*

If we take into account the previous remark, we get:

$$\Gamma_{ij}^k = \Bigg[\left(\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)-2p-\frac{(1-2p)}{2}\right)\left(\alpha_i(x)\delta_j^k + \alpha_j(x)\delta_i^k - \alpha^k(x)g_{ij}\right) \\ + \frac{1}{p(1-p)}(p(1-p))^2\frac{\partial\alpha_i(x)}{\partial x_j}\alpha^k(x)\Bigg]$$

$$\Gamma_{ij}^k = \Bigg[\left(\left(W_2\sigma\left(W_1x+b_1\right)+b_2\right)-p-\frac{1}{2}\right)\left(\alpha_i(x)\delta_j^k + \alpha_j(x)\delta_i^k - \alpha^k(x)g_{ij}\right) \\ + p(1-p)\frac{\partial\alpha_i(x)}{\partial x_j}\alpha^k(x)\Bigg]$$

and for the derivative of the metric we get:

$$\begin{aligned}\frac{\partial g_{ij}}{\partial x_l} &= \frac{1}{p(1-p)}\Big[-J_l\,(1-2p)g_{ij} \\ &\qquad + 2\left((W_2\sigma\,(W_1x+b_1)+b_2)-2p\right)J_l g_{ij} \\ &\qquad\qquad + (a(x))^2\left(\frac{\partial\alpha_i(x)}{\partial x_l}\alpha_j(x)+\alpha_i(x)\frac{\partial\alpha_j(x)}{\partial x_l}\right)\Big] \\ &= \frac{1}{p(1-p)}\Big[\left(2\,(W_2\sigma\,(W_1x+b_1)+b_2)-2p-1\right)J_l g_{ij} \\ &\qquad\qquad + (a(x))^2\left(\frac{\partial\alpha_i(x)}{\partial x_l}\alpha_j(x)+\alpha_i(x)\frac{\partial\alpha_j(x)}{\partial x_l}\right)\Big] \\ &= \Big[\left(2\,(W_2\sigma\,(W_1x+b_1)+b_2)-2p-1\right)\alpha_l(x)g_{ij} \\ &\qquad + \left(\frac{\partial\alpha_i(x)}{\partial x_l}J_j + J_i(x)\frac{\partial\alpha_j(x)}{\partial x_l}\right)\Big].\end{aligned}$$

B. Notions of Riemannian geometry

Let $\mathcal{M}=(M,g)$ be a real Riemannian manifold of dimension n.

Definition 8 (length of a path) *Let $\gamma : [0,1] \to M$ be a C^1 path. The* lenght *of γ, denoted $l(\gamma)$ is defined by*

$$l(\gamma)=\int_0^1 g\left(\gamma\,(t)\,,\gamma'\,(t)\,,\gamma'\,(t)\right)^{1/2}dt$$

Definition 9 (geodesic) *Let $p,\ q$ be two points of M. The C^1 path $\gamma:[0,1]\to M$ is said to be a* geodesic *between p and q if:*

$$\begin{cases}\gamma(0)=p,\ \gamma(1)=q \\ l(\gamma)=\inf\left\{l(\theta),\ \theta\in C^1\left([0,1),\ M\right),\ \theta(0)=p,\ \theta(1)=q\right\}\end{cases}$$

Definition 10 (geodesic distance) *The Length of a geodesic between p and q is called the* geodesic distance*, denoted $d(p,q)$.*

In what follows, the Levi-Civita connexion of $\mathcal{M}$ will be denoted $\mathring{\nabla}$.

Definition 11 *Let $\gamma:[0,1]\to M$ be a C^2 path. It it said to be a geodesic of $\mathring{\nabla}$ is, for each $t\in]0,1[$ the following holds:*

$$\mathring{\nabla}_{\dot\gamma(t)}\dot\gamma(t)=0 \tag{10}$$

The differential equation 10 translates in local coordinates to:

$$\frac{\partial^2\gamma^k}{\partial t^2}(t)+\Gamma^k_{ij}\left(\gamma\,(t)\right)\frac{\partial\gamma^i}{\partial t}(t)\frac{\partial\gamma^j}{\partial t}(t)=0 \tag{11}$$

For an initial point $p=\gamma(0)$, Cauchy-Lipschitz theorem shows that ther is a unique local solution to Equation 10.

Proposition B.1 *Let $p \in M$. There exist $\varepsilon > 0$ such that for all $v \in T_pM$, $\|v\| < \varepsilon$, there exist a unique geodesic $\gamma : [0,1] \to M$ such that $\gamma(0) = p,\ \gamma'(0) = v$. The function which to such v associates $\gamma(1)$ with γ the geodesic of $\mathring{\nabla}$ such that $\gamma(0) = p,\ \gamma'(0) = v$ is called the* exponential map *and denoted* $\exp_p$.

Proposition B.2 *Let $v \in T_pM$, $\|v\| < \varepsilon$ and let $q = \exp_p(v)$. Then $\gamma : t \in [0,1] \mapsto \exp_p(tv)$ is a geodesic between p and q. Besides, $l(\gamma) = \|v\|$.*

Remark 13 *Note that $\|v\| = g\left(p; v, v\right)^{1/2}$ is the riemannian norm and not the Euclidean norm.*

Definition 12 *$d\exp_p(0) = Id$ and thus the exponential map is a local diffeomorphism. Around each point, $\exp_p$ defines a chart of M. The local coordinates we get are called the* normal coordinates *at p.*

Definition 13 (logarithm map) *Let $p \in M$ and $\varepsilon > 0$ such that the exponential map is defined in $B(0,\varepsilon)$. For all $q \in M$ such that $d(p,q) < \varepsilon$ we set:*

$$\log_p(q) = v,\ \ \exp_p v = q.$$

Remark 14 *One can compute the logarithm by solving the following differential system:*

$$\begin{cases} \frac{\partial^2 \gamma^k}{\partial t^2}(t) + \Gamma_{ij}^k\left(\gamma(t)\right) \frac{\partial \gamma^i}{\partial t}(t) \frac{\partial \gamma^j}{\partial t}(t) = 0 \\ \gamma(0) = p,\ \gamma(1) = q \end{cases} \tag{12}$$

Proposition B.3 *In normal coordinates at $p \in M$, geodesics with origin p are strait lines going through the origin.*

Definition 14 (parallel transport) *Let $v \in T_pM$ and $q = \exp_p x$. The geodesic between p and q is $\gamma : t \in [0,1] \mapsto tx$ in normal coordinates. Besides, the linear differential equation:*

$$\mathring{\nabla}_{\dot{\gamma}(t)} X(t) = 0,\ \ X(0) = v$$

has a solution on $[0,1]$ called the parallel transport *of v.*

Parallel transport allows to go from a tangent vector at $q = \exp_p x$ to a tangent vector at p.