COMPUTATIONAL OPTIMAL TRANSPORT PROJECT: KERNELIZED WASSERSTEIN NATURAL GRADIENT

Léon Zheng
M2 MVA "Mathématiques, Vision, Apprentissage"
ENS Paris-Saclay, France
leon.zheng@polytechnique.org

ABSTRACT

A machine learning optimization problem can be formulated as an optimization of a loss function over a family of parameterized probability distributions. While stochastic gradient descent methods perform well in these optimization tasks, they lack robustness to ill-conditionning of the problem. Natural gradient directly operates on the manifold of the parametric model and therefore has properties of robustness to reparametrizations of the problem. However, computing the exact natural gradient requires to invert the metric tensor, which is not tractable in problems with millions of parameters like in deep neural network. The paper studied in this project proposes to estimated the Wasserstein natural gradient, induced by the Wasserstein 2 metric, using kernel estimators. The authors formulate a functional quadratic optimization problem over a reproducing Hilbert kernel space, for which the solution is an approximation of the Wasserstein natural gradient. This solution is then estimated from samples with Nyström method, a general method for low-rank approximations of kernels. The proposed estimator has theoretical guarantees of convergence. It is also accurate enough and computationally manageable for practical use in neural network optimization on classifications tasks with `Cifar10` and `Cifar100` data sets. This project shows in particular that the choice of the kernel used in the estimation is important in practice.

1 Introduction

1.1 Presentation of the problem

In [Arbel et al., 2019], we consider the optimization of some cost functional over a parametric family of probability distributions, in the context of machine learning problems.

Gradient descent method Many state-of-the-art methods rely on Stochastic Gradient Descent (SGD) like AdaGrad [Duchi et al., 2010], RMSProp [Hinton et al., 2012], and Adam [Kingma and Ba, 2014]. These methods are generally effective, but are sensitive to the curvature of the optimization objective. As described in [Boyd and Vandenberghe, 2004], in the case of convex optimization, gradient descent is a first order method using the local gradient at each iteration to find the steepest descent direction. Depending on how well the problem is conditioned, this descent direction doesn't point exactly towards the minimum of the objective function, which can lead to inefficient parameter trajectory and slow convergence.

Natural gradient For the optimization of some cost functional over a parametric family of probability distributions, natural gradient descent [Amari, 1998] is an optimization method which takes into account the local curvature of the manifold of the parametric probability distributions, and is no more sensitive to the conditioning of the problem. The idea is to overcome the difficulties of ill-conditioning by adapting the descent direction to the curvature of the manifold of distributions, for faster convergence. Given a metric between probability distributions, we can pull-back this metric in the parameter space, through the mapping from the parameters space to the probability distributions space given by the considered parametric model. It is then possible to express a gradient preconditioned to this pulled-back metric, which is called the natural gradient. In a natural gradient descent method, we use this natural gradient as the steepest descent direction.

Classical choice of the metric in the distribution space is the Fisher-Rao metric, which induces the Fisher-Rao natural gradient [Amari, 1998]. In [Arbel et al., 2019], the considered metric is the Wasserstein 2 metric, which induces the Wasserstein natural gradient [Chen and Li, 2018].

Estimating the natural gradient Although having a lot a theoretical advantages, using natural gradient in practice is challenging, because each parameter update needs to invert the metric tensor. This becomes intractable when the number of parameters q is very large, up to one million for deep neural networks, since inverting a matrix of size q without any structure requires a cost of $\mathcal{O}(q^3)$. A main issue to use the natural gradient in practice is to find a way to compute an approximation of this natural gradient, with a good trade-off between accuracy and computational cost. The more specific problem studied in [Arbel et al., 2019] is to find an efficient way to estimate with samples the Wasserstein natural gradient induced by the Wasserstein 2 metric.

1.2 Related works

Newton's method In the case of convex optimization problems, Newton's method is the general method to make an optimizer invariant to the conditioning of the problem, as it is explained in [Boyd and Vandenberghe, 2004]. At each step, the local Hessian of the objective function is positive, so we can compute the gradient induced by the metric defined by this positive matrix, and use it as the descent direction. With this kind of quadratic approximation, this optimizer is approximately adapted to the curvature of the optimization objective. However, this method requires the computation of the Hessian and its inverse, which is a strong requirement.

Approximation with Kronecker products Kronecker-factored Approximate Curvature (KFAC) [Martens and Grosse, 2015] is an efficient method to approximate the Fisher-Rao natural gradient in neural networks. The method relies on an efficient approximation of the Fisher information matrix so that it can be efficiently inverted. This approximation is derived from two steps. In the first one, the Fisher information matrix is decomposed in blocks, each of which corresponds to all the weights of a given layer. These blocks are then approximated as the Kronecker product of smaller matrices, which is shown to be equivalent to make some assumptions about the statistics of the network's gradient. In the second step, the obtained matrix is approximated as a matrix having an inverse which is either block-diagonal or block-tridiagonal, in order to perform inverse.

In [Martens and Grosse, 2015], practical performances of KFAC has been tested for autoencoder problems on MNIST, CURVES, and FACES data sets, and compared to the stochastic gradient descent (SGD) method with momentum based on Nesterov's accelerated gradient. When combined with a form of momentum and an increasing schedule for the mini-batch size m, the KFAC method outperforms SGD with momentum. The approximations made in the method are good enough to solve the tasks efficiently, with less overall computational time.

One possible improvement of KFAC is Eigenvalue-corrected Kronecker Factorization (EKFAC) [George et al., 2018]. KFAC method approximates the Fisher information matrix by approximating it as a Kroenecker product and considering the eigenspectrum of this approximation. However, this approximated eigenspectrum is not guaranteed to have the same scale as the eigenspectrum of the Fisher information matrix. EKFAC corrects the scaling and achieves a better approximation of the Fisher information matrix. The authors gave also a proof that EKFAC yields a more accurate estimate than KFAC, in the sense of the Frobenius norm. Experiments on deep autoencoders show that EKFAC is still computationally manageable and quite accurate, and is a good improvement of KFAC, still outperforming SGD with momentum.

Proximal method, dual formulation Instead of estimating the inverse of the metric tensor, [Li et al., 2019] uses proximal methods as an alternative way to express the update at each natural gradient descent iteration. At each iteration, an optimization problem is solved to find the minimum of the loss function penalized by a proximity term which favors close points to the current one. This proximity term is constructed from the pulled-back metric. Using dual formulation of this optimization problem, [Li et al., 2019] proposes an approximation its solution when considering the Fisher-Rao metric and the Wasserstein metric. This method avoids the computation of the natural gradient, but requires to solve an additional optimization problem. The quality of the solver depends on the accuracy of this optimization problem.

1.3 Contributions

[Arbel et al., 2019] proposes an estimator for the Wasserstein natural gradient using kernel estimators, which is a novel idea, and different from methods relying on Kronecker product approximation and eigenspectrum decomposition like KFAC, EKFAC, or proximal methods like in [Li et al., 2019].

Denote Ω, the space on which are defined the considered probability distributions. By exploiting Legendre duality for metrics, the Wasserstein natural gradient can be expressed as the solution of a convex functional optimization problem over $C_c^\infty(\Omega)$, the set of smooth and compactly supported functions on Ω. Then, this functional optimization problem is restricted over a reproducing kernel Hilbert space (RKHS), whose unique solution is called the kernelized Wasserstein natural gradient (KWNG). Finally, [Arbel et al., 2019] constructs an estimator of this exact solution using low-rank approximations of kernels, which can be computed with samples of the data set, and is guaranteed to converge to the exact solution with a given convergence rate. Therefore, in this method, there is no need to inverse the Wasserstein metric tensor. Instead, computing the estimator requires a pseudo-inverse of a matrix of size $M \leq N$, with N the number of samples.

Experiences in [Arbel et al., 2019] illustrated the accuracy of the proposed estimator for the Wasserstein natural gradient on easy examples, in the limit of large number of samples. It is shown experimentally that the trajectory of Wasserstein natural gradient descent is well approximated when using the estimator derived from the paper's method. [Arbel et al., 2019] also showed performances of natural gradient descent methods using the estimator of KWNG as the steepest descent direction, and compared to gradient descent methods, which are sensitive to the conditioning, and other natural gradient methods induced by the Fisher-Rao metrics, like KFAC [Martens and Grosse, 2015] and EFKAC [George et al., 2018]. The estimator is consistent enough to be used in classification tasks like `Cifar10` and `Cifar100` with a neural networks with a relatively high number of parameters (one convolutional layer with a kernel size followed by 8 residual blocks and a fully connected layer). The author also demonstrated experimentally the robustness of the estimator of KWNG to ill-conditioned case.

The Wasserstein metric has the advantage of being well defined even when the model doesn't admit a density, which is not the case for the Fisher-Rao metric. However, it is not explained in the paper why theoretically the descent method using the estimated kernelized Wasserstein natural gradient should perform better than when using KFAC or EKFAC, although the experiments showed better results for KNWG.

The method proposed by [Arbel et al., 2019] to estimate the Wasserstein natural gradient from the variational formulation derived using duality for metrics and kernel methods is a general method which can be applied to other metrics, like the Fisher-Rao metric. One interesting future work left by the authors is to derive an estimator of the Fisher-Rao natural gradient using this same method, study the trade-off between its accuracy and its computational cost, and compare it with other estimators of the Fisher-Rao metric like KFAC or EKFAC. If this estimator is consistent, we can then compare the performance of a descent method using this estimator of the Fisher-Rao natural gradient to a descent method using the estimated KWNG.

1.4 Project work

The work in this project includes:

1. a presentation of the method of [Arbel et al., 2019] to estimate the Wasserstein natural gradient, and a discussion about this method;
2. an implementation of the proposed estimator, a discussion about the practical considerations of this method, and a reproduction of the paper's experiments about the performance of descent methods using the estimated KWNG on some toy examples;
3. a derivation of an estimator of the Fisher-Rao natural gradient using the method proposed by [Arbel et al., 2019], and an implementation of this estimator on a toy example.

2 Problem formulation

Let $\Theta \subset \mathbb{R}^q$ an open parameter space of dimension q, and $\Omega \subset \mathbb{R}^d$ an open sample space of dimension d. Let ν a latent distribution defined over a latent space $\mathcal{Z}$. For each parameter $\theta \in \Theta$, we consider a deterministic function $z \mapsto h_\theta(z)$ from the latent space $\mathcal{Z}$ to the sample space Ω. This defines an implicit parametric model $\mathcal{P}_\Theta = \{\rho_\theta := (h_\theta)_\# \nu \mid \theta \in \Theta\}$, where $(h_\theta)_\# \nu$ is the push-forward of ν by the function h_θ. Consider also some cost function $\rho \mapsto \mathcal{F}(\rho)$ over the parametric model $\mathcal{P}_\Theta$. Define the loss function $\mathcal{L} : \theta \mapsto \mathcal{F}(\rho_\theta)$. The learning problem is then formulated as:

$$\min_{\theta \in \Theta} \mathcal{L}(\theta) := \mathcal{F}(\rho_\theta) \tag{1}$$

Consider a metric g on a probability distributions space containing the parametric model $\mathcal{P}_\Theta$. The matrix representing the metric on the space of parameters Θ pulled-back from the metric g through the parametric model $\mathcal{P}_\Theta$ is called the information matrix, denotes $G(\theta) \in \mathbb{R}^{q \times q}$ for all $\theta \in \Theta$.

In this project, we will consider the Fisher-Rao metric and the Wasserstein 2 metric. These two metrics induce in the parameter space the Fisher information matrix [Amari, 1985] and the Wasserstein information matrix [Chen and Li, 2018].

Definition 1 (Fisher information matrix)**.** *Assume $\theta \mapsto \rho_\theta$ is differentiable for all $x \in \Omega$, and that $\int \frac{\|\nabla \rho_\theta(x)\|^2}{\rho_\theta} dx < \infty$. Then the Fisher information matrix is defined as the pull-back of the Fisher-Rao metric g^F:*

$$G_F(\theta)_{ij} = g^F_{\rho_\theta}(\partial_i \rho_\theta, \partial_j \rho_\theta) := \int \frac{\partial_i \rho_\theta(x)}{\rho_\theta(x)} \frac{\partial_j \rho_\theta(x)}{\rho_\theta(x)} \rho_\theta(x) dx. \tag{2}$$

Definition 2 (Wasserstein information matrix)**.** *The Wasserstein information matrix is defined as the pull-back of the Wasserstein 2 metric g^W:*

$$G_W(\theta)_{ij} = g^W_{\rho_\theta}(\partial_i \rho_\theta, \partial_j \rho_\theta) := \int \phi_i(x)^T \phi_j(x) \rho_\theta(x) dx,$$

where ϕ_i are vectors valued functions on Ω that are solutions to the partial differential equations with Neumann boundary condition:

$$\partial_i \rho_\theta = -div(\rho_\theta \phi_i), \qquad \forall i \in [\![1, q]\!].$$

Moreover, ϕ_i are required to be in the closure of the set of gradients of smooth and compactly supported functions in $L_2(\rho_\theta)^d$. In particular, when ρ_θ has a density, $\phi_i = \nabla_x f_i$, for some real valued function f_i on Ω.

Given an information matrix $G(\theta)$, we define its corresponding natural gradient $\nabla^G \mathcal{L}(\theta)$ as the solution of the following minimization problem:

$$\nabla^G \mathcal{L}(\theta) := -\arg\min_{u \in \mathbb{R}^q} \mathcal{M}(u) + \frac{1}{2} u^T G(\theta) u \tag{3}$$

where $\mathcal{M}(u) = \mathcal{L}(\theta) + \nabla \mathcal{L}(\theta)^T u$ is the first order expansion of the loss function at θ_t, and $\nabla \mathcal{L}$ is the Euclidean gradient of the loss function $\mathcal{L}$. The solution of (3) can then be expressed in closed form using the inverse of the information matrix and the Euclidean gradient: $\nabla^G \mathcal{L}(\theta) = G(\theta)^{-1} \nabla \mathcal{L}(\theta)$. Finally, the natural gradient descent method for optimizing (1) consists in the following parameters update rule:

$$\theta_{t+1} = \theta_t - \gamma_t \nabla^G \mathcal{L}(\theta_t) \tag{4}$$

where $(\theta_t)_t$ is the sequence of parameters produced by the natural gradient descent, $(\gamma_t)_t$ is the sequence of step size.

The problem is to estimate efficiently the natural gradient at each iteration step. The goal is to find an estimator which has a good trade-off between accuracy and computational cost. In particular, computing the inverse of the information matrix at each step of the natural gradient descent should be avoided. In [Arbel et al., 2019], a method based on kernel estimators is proposed to estimate the Wasserstein natural gradient $\nabla^W \mathcal{L}(\theta)$, which is the natural gradient induced by the Wasserstein information matrix. We will see in this report that this method can also be used to derive an estimator of the Fisher-Rao natural gradient $\nabla^F \mathcal{L}(\theta)$, which is the natural gradient induced by the Fisher information matrix.

3 Kernelized Wasserstein natural gradient

This section presents the method to construct an estimator of the Wasserstein natural gradient proposed by [Arbel et al., 2019]. Most elements in the following paragraphs come from the paper.

3.1 Legendre duality for metrics

To construct this estimator, this first step in [Arbel et al., 2019] is to express the Wasserstein information matrix as the solution of a functional optimization problem, by exploiting Legendre duality for metrics.

Proposition 1. *Under technical assumptions presented in [Arbel et al., 2019], the Wasserstein information matrix satisfies, for all $\theta \in \Theta$:*

$$\frac{1}{2} u^T G_W(\theta) u = \sup_{\substack{f \in C_c^\infty(\Omega) \\ \int f(x) d\rho_\theta(x) = 0}} \nabla \rho_\theta(f)^T u - \frac{1}{2} \int \|\nabla_x f(h_\theta(z))\|^2 d\nu(z) \tag{5}$$

The gradient $\nabla \rho_\theta(f)$ should be understood in the distribution sense, like in the following definition.

Definition 3. *Given a parametric family $\mathcal{P}_\Theta$ of probability distributions, the distributional gradient $\nabla \rho_\theta$ at point θ of ρ_θ is defined as the linear map $\nabla \rho_\theta : C_c^\infty(\Omega) \to \mathbb{R}^q$ whose components are given by:*

$$(\nabla \rho_\theta(f))_i = \lim_{\epsilon \to 0} \epsilon^{-1} \left(\int f(x) d\rho_{\theta + \epsilon e_i}(x) - \int f(x) d\rho_\theta(x) \right)$$

where $(e_i)_{1 \le i \le q}$ is an orthonormal basis of $\mathbb{R}^q$.

3.2 Kernel methods

The second step in the method of [Arbel et al., 2019] is to restrict the functional optimization problem in (5) to a reproducing kernel Hilbert space $\mathcal{H}$, with its inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and its norm $\| \cdot \|_{\mathcal{H}}$. Its kernel $k : \Omega \times \Omega \mapsto \mathbb{R}$, where $k(x, \cdot) \in \mathcal{H}$ for all $x \in \Omega$, satisfies the reproducing kernel property: $\forall f \in \mathcal{H}, \forall x \in \Omega, f(x) = \langle f, k(x, \cdot) \rangle_{\mathcal{H}}$. Combining the minimization problem (3) with the functional optimization problem (5) restricted to $\mathcal{H}$, and adding some regularization terms, we obtain the following saddle problem for all $\theta \in \Theta$:

$$\min_{u \in \mathbb{R}^q} \sup_{\substack{f \in \mathcal{H} \\ \int f(x) d\rho_\theta(x) = 0}} \mathcal{U}_\theta(f)^T u - \frac{1}{2} \int \|\nabla_x f(x)\|^2 d\rho_\theta(x) + \frac{1}{2}(\epsilon u^T D(\theta) u - \lambda \|f\|^2_{\mathcal{H}}) \tag{6}$$

where $\mathcal{U}_\theta(f) := \nabla \mathcal{L}(\theta) + \nabla \rho_\theta(f)$, $\epsilon > 0$, $\lambda > 0$, and $D(\theta)$ is a diagonal matrix in $\mathbb{R}^{q \times q}$ with positive values. The solution of this saddle problem defines the kernelized Wasserstein natural gradient, and it is written $\widetilde{\nabla}^W \mathcal{L}(\theta)$. By exchanging the order of the supremum and minimum of this saddle problem, the kernelized Wasserstein natural gradient satisfies the following proposition from [Arbel et al., 2019].

Proposition 2. *The kernelized Wasserstein natural gradient is given by:*

$$\widetilde{\nabla}^W \mathcal{L}(\theta) = \frac{1}{\epsilon} D(\theta)^{-1} \mathcal{U}_\theta(f^*), \tag{7}$$

where f^* is the unique solution to the quadratic optimization problem:

$$\inf_{\substack{f \in \mathcal{H} \\ \int f(x) d\rho_\theta(x) = 0}} \mathcal{J}(f) := \int \|\nabla_x f(x)\|^2 d\rho_\theta(x) + \frac{1}{\epsilon} \mathcal{U}_\theta(f)^T D(\theta)^{-1} \mathcal{U}_\theta(f) + \lambda \|f\|^2_{\mathcal{H}}. \tag{8}$$

The proof of this proposition isn't given in the paper, but it is similar to the proof of Proposition 5 in this report.

3.3 Nyström methods

The third step in [Arbel et al., 2019] is to derive an estimator for the solution of (8) which can be computed efficiently using Nyström methods, which are general methods for low-rank approximation of kernels. Consider N samples $(Z_n)_{1 \leq n \leq N}$ from the latent distribution ν, which are used to produce N samples $(X_n)_{1 \leq n \leq N}$ from ρ_θ using the map h_θ, i.e., $X_n = h_\theta(Z_n)$. By introducing basis points $(Y_m)_{1 \leq m \leq M}$ uniformly drawn from $(X_n)_{1 \leq n \leq N}$ with $M \leq N$, and M indices $(i_m)_{1 \leq m \leq M}$ uniformly drawn from $[\![1, d]\!]$, we can consider a finite dimensional subspace $\mathcal{H}_M$ of $\mathcal{H}$, called the Nyström subspace:

$$\mathcal{H}_M := span\{x \mapsto \partial_{i_m} k(Y_m, x) \mid 1 \leq m \leq M\}. \tag{9}$$

An estimator for the kernelized Wasserstein natural gradient will then be:

$$\widehat{\nabla^W \mathcal{L}(\theta)} = \frac{1}{\epsilon} D(\theta)^{-1} \hat{\mathcal{U}}_\theta(\widehat{f^*}) \tag{10}$$

where $\widehat{f^*}$ is the unique solution to the quadratic optimization problem:

$$\widehat{f^*} := \arg \min_{\substack{f \in \mathcal{H}_M \\ \int f(x) d\rho_\theta(x) = 0}} \frac{1}{N} \sum_{n=1}^{N} \|\nabla_x f(X_n)\|^2 + \frac{1}{\epsilon} \hat{\mathcal{U}}_\theta(f)^T D(\theta)^{-1} \hat{\mathcal{U}}_\theta(f)^T + \lambda \|f\|^2_{\mathcal{H}} \tag{11}$$

with $\hat{\mathcal{U}}_\theta(f) = \widehat{\nabla \mathcal{L}(\theta)} + \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta h_\theta(Z_n)^T \nabla_x f(X_n)$ and $\widehat{\nabla \mathcal{L}(\theta)}$ an estimator of the Euclidean gradient $\nabla \mathcal{L}(\theta)$. The matrix $\nabla_\theta h_\theta(Z_n)$ is the Jacobian matrix of $\theta \mapsto \nabla_\theta h_\theta(Z_n)$. [Arbel et al., 2019] provides a closed form expression for the proposed estimator of the kernelized Wasserstein natural gradient.

Proposition 3. *The estimator given in (10) of the kernelized Wasserstein natural gradient satisfies:*

$$\widehat{\nabla^W \mathcal{L}(\theta)} = \frac{1}{\epsilon} \left(D(\theta)^{-1} - D(\theta)^{-1} T^T \left(T D(\theta)^{-1} T^T + \lambda \epsilon K + \frac{\epsilon}{N} C C^T \right)^\dagger T D(\theta)^{-1} \right) \widehat{\nabla \mathcal{L}(\theta)} \tag{12}$$

where C and K are matrices in $\mathbb{R}^{M \times Nd}$ and $\mathbb{R}^{M \times M}$ given by:

$$C_{m,(n,i)} = \partial_{i_m} \partial_{i+d} k(Y_m, X_n), \quad K_{m,m'} = \partial_{i_m} \partial_{i_{m'}+d} k(Y_m, Y_{m'}), \tag{13}$$

while T is a matrix in $\mathbb{R}^{M \times q}$ obtained as the Jacobian of $\theta \mapsto \tau(\theta)$ where:

$$\forall m \in [\![1, M]\!], \quad (\tau(\theta))_m = \frac{1}{N} \sum_{n=1}^{N} \partial_{i_m} k\left(Y_m, h_\theta(Z_n)\right). \tag{14}$$

The notation $\partial_i k(a,b)$ is the partial derivative of $a \mapsto k(a,b)$ with respect to a_i, the i-th coordinate of $a \in \Omega$, when $b \in \Omega$ is fixed. Similarly, the notation $\partial_{j+d} k(a,b)$ is the partial derivative of $b \mapsto k(a,b)$ with respect to b_j. In practice, the choice of the diagonal matrix $D(\theta)$ is $(D(\theta))_i = \|T_{\cdot,i}\|^2$ for $i \in [\![1,d]\!]$.

3.4 Theoretical guarantees

[Arbel et al., 2019] offers theoretical guarantees about the behavior of this estimator in the limit of large N and M. We assume a well-specified case where the functions $(\phi_i)_i$ involved in Definition 2 can be expressed as gradient of functions $f_i \in \mathcal{H}$, and also some technical assumptions about the parametric model $\mathcal{P}_\Theta$ and the kernel k. Then, Theorem 5 in [Arbel et al., 2019] gives the following convergence rate of the proposed estimator of KWNG in Proposition 3. For N large enough, $M \sim dN^{-\frac{1}{2b+1}} \log(N)$, $\lambda \sim N^{-\frac{1}{2b+1}}$ and $\epsilon \leq N^{-\frac{b}{2b+1}}$, it holds with probability $1-\delta$ for $0 \leq \delta \leq 1$ that:

$$\|\widehat{\nabla^W \mathcal{L}(\theta)} - \nabla^W \mathcal{L}(\theta)\|^2 = \mathcal{O}(N^{-\frac{2b}{2b+1}}) \tag{15}$$

where $b = \min(1, \alpha + \frac{1}{2})$ and $\alpha \geq 0$ is a parameter which characterizes the smoothness of f_i and controls the statistical complexity of the estimation problem.

If we consider the worst case, where $\alpha = 0$, the proposed estimator need at most $M \sim d\sqrt{N}\log(N)$ to achieve a convergence rate of $N^{-\frac{1}{2}}$. This gives an idea of the choice of the number of basis points for practical uses with a good trade-off between accuracy and computational cost. We will choose for example in practice $M = \left\lfloor d\sqrt{N} \right\rfloor$.

3.5 Discussion about the method

Hypotheses In order to perform this method to estimate the KWNG, we need to know how to: (i) sample latent random variables Z following the latent distribution ν; (ii) compute the Jacobian of $\theta \mapsto h_\theta(z)$ for all $z \in \mathcal{Z}$; (iii) compute the partial derivatives of the kernel k. In particular, it doesn't require strong assumptions on the parametric model $\mathcal{P}_\Theta$: for $\rho_\theta \in \mathcal{P}_\Theta$, $\theta \mapsto \rho_\theta$ doesn't need to be differentiable, which is not the case when we want to use Fisher-Rao natural gradient. Instead, we only need less restrictive assumptions on $\theta \mapsto h_\theta$.

Choice of the kernel With Theorem 5 in [Arbel et al., 2019] about the consistency of the estimator, the choice of the kernel k is only limited by the assumptions of the theorem:

1. k is twice continuously differentiable on $\Omega \times \Omega$;
2. for all $\theta \in \Theta$, it holds that $\int \partial_i \partial_{i+d} k(x,x) d\rho_\theta(x) < \infty$ for all $1 \leq i \leq d$;
3. the quantity $\sup_{x \in \Omega, 1 \leq i \leq d} \partial_i \partial_{i+d} k(x,x)$ is finite.

Examples of possible kernels for the estimator are:

- the Gaussian kernel $k(a,b) = \exp\left(-\frac{\|a-b\|^2}{2\sigma^2}\right)$, with $\sigma > 0$;
- the sigmoid kernel $k(a,b) = \tanh\left((\alpha a^T b + c)\right)$.

Examples of kernels not satisfying the previous assumptions are:

- the Laplacian kernel $k(a,b) = \exp\left(-\frac{\|a-b\|}{\sigma}\right)$, with $\sigma > 0$, because it is not twice differentiable everywhere, for example when $a = b$;
- the polynomial kernel $k(a,b) = (\alpha a^T b + c)^\gamma$, with γ an integer, because it doesn't satisfy the third assumption.

In theory, any kernel that satisfies the previous assumptions can be used for computing the estimator in Proposition in 3, even though it is not clear in [Arbel et al., 2019] how the kernel should be chosen in practice. Experiments in Section 5 will show that the choice of the kernel isn't easy, and we have to fine-tune the parameters of the kernel to get a right estimation of KWNG.

Computational cost The method using kernel estimators avoids computing the inverse of the whole information matrix, but the cost of computing the estimator of KWNG at each iteration in Proposition 3 is controlled by the number of basis points M. Indeed, the main contributions cost in the computation of (12) are:

- the cost of CC^T where $C \in \mathbb{R}^{M \times Nd}$ which is $\mathcal{O}(M^2 dN)$;

- the cost of inverting $B = \left(TD(\theta)^{-1}T^T + \lambda\epsilon K + \frac{\epsilon}{N}CC^T\right) \in \mathbb{R}^{M\times M}$ which is $\mathcal{O}(M^3)$;
- the cost of $TB^{-1}T^T$ when B^{-1} is already computed and $T \in \mathbb{R}^{M\times q}$ which is $\mathcal{O}(qM^2)$.

As discussed in Section 3.4, the number of basis points M can be relatively small, since $M = \left\lfloor d\sqrt{N} \right\rfloor$ is enough for achieving convergence of the estimator. In a neural network training, N corresponds to the number of samples in a mini-batch. Therefore, the estimator of KWNG is computationally manageable, as long as the dimension d and the number of parameters q are not too large. For a deep neural network architecture, the number of parameters is $q \leq 10^6$, although Figure 3 and 4 in [Arbel et al., 2019] showed good performances of the estimated KWNG in deep neural networks.

4 Kernelized Fisher-Rao natural gradient

The method in Section 3 is a general method to derive an estimator of a natural gradient, as long as we can express the natural gradient as the solution of a functional optimization problem. In this section, we derive an estimator of the Fisher-Rao natural gradient based on the method described in Section 3. We start with the following result from [Arbel et al., 2019], which is the variational formulation of the Fisher information matrix using Legendre duality for metrics.

Proposition 4. *Under the same assumptions as in Definition 1, the Fisher information matrix satisfies:*

$$\frac{1}{2}u^T G_F(\theta)u = \sup_{\substack{f\in C_c^\infty(\Omega) \\ \int f(x)d\rho_\theta(x)=0}} \nabla\rho_\theta(f)^T u - \frac{1}{2}\int f(x)^2\rho_\theta(x)dx. \tag{16}$$

We consider again a reproducing kernel Hilbert space $\mathcal{H}$, with the same notation as in Section 3. The kernel associated to this RKHS is called k. We consider the following saddle problem:

$$\min_{u\in\mathbb{R}^q} \sup_{\substack{f\in\mathcal{H} \\ \int f(x)d\rho_\theta(x)=0}} L(u,f) := \mathcal{U}_\theta(f)^T u - \frac{1}{2}\int f(x)^2 d\rho_\theta(x) + \frac{1}{2}(\epsilon u^T D(\theta)u - \lambda\|f\|_{\mathcal{H}}^2) \tag{17}$$

where $\mathcal{U}_\theta(f) := \nabla\mathcal{L}(\theta) + \nabla\rho_\theta(f)$, $\epsilon > 0$, $\lambda > 0$, and $D(\theta)$ is a diagonal matrix in $\mathbb{R}^{q\times q}$ with positive values. The solution of (17) is called the kernelized Fisher-Rao natural gradient (KFRNG). By inverting the order of the supremum and the minimum, we obtain the following expression for the kernelized Fisher-Rao natural gradient.

Proposition 5. *The kernelized Fisher-Rao natural gradient is given by:*

$$\widetilde{\nabla}^F\mathcal{L}(\theta) = \frac{1}{\epsilon}D(\theta)^{-1}\mathcal{U}_\theta(f^*), \tag{18}$$

where f^* is the unique solution to the quadratic optimization problem:

$$\inf_{\substack{f\in\mathcal{H} \\ \int f(x)d\rho_\theta(x)=0}} \mathcal{K}(f) := \int f(x)^2 d\rho_\theta(x) + \frac{1}{\epsilon}\mathcal{U}_\theta(f)^T D(\theta)^{-1}\mathcal{U}_\theta(f) + \lambda\|f\|_{\mathcal{H}}^2. \tag{19}$$

Proof. The particular choice of the regularization term $\frac{1}{\epsilon}\mathcal{U}_\theta(f)^T D(\theta)^{-1}\mathcal{U}_\theta(f) + \lambda\|f\|_{\mathcal{H}}^2$ allow to use a version of the minimax theorem from [Ekeland and Téman, 1976], Proposition 2.3, Chapter VI. Indeed, the hypotheses needed for using this theorem are verified. $\mathbb{R}^q$ is convex, closed and non empty. The set $\mathcal{H}' = \{f \in \mathcal{H} \mid \int f(x)d\rho_\theta(x) = 0\}$ is convex by linearity of the intergral, closed as the inverse image of $\{0\}$ by a continuous function, and non empty since it contains the function 0. For all $u \in \mathbb{R}^q$, $f \mapsto L(u,f)$ is continuous and concave, since $f \mapsto \int f^2 d\rho_\theta$ is a quadratic function, and $\|\cdot\|_{\mathcal{H}}$ is convex as a norm. For all $f \in \mathcal{H}'$, $u \mapsto L(u,f)$ is a quadratic form, so it is convex and continuous. Finally, there exists $f_0 \in \mathcal{H}'$ such that $\lim_{\|u\|\to\infty} L(u,f_0) = +\infty$: we can take for example $f_0 = 0$.

For a fixed function $f \in \mathcal{H}'$, the infimum of $u \mapsto L(u,f)$ is reached at $-\frac{1}{2}\mathcal{U}_\theta(f)^T D(\theta)^{-1}\mathcal{U}_\theta(f)$ since we recognize a minimization problem of a quadratic form, and the minimizer $-\frac{1}{\epsilon}D(\theta)^{-1}\mathcal{U}_\theta(f)$ is unique.

Therefore:

$$\begin{aligned}
\min_{u\in\mathbb{R}^q}\sup_{f\in\mathcal{H}'} L(u,f) &= \sup_{f\in\mathcal{H}'}\inf_{u\in\mathbb{R}^q} L(u,f) \\
&= \sup_{f\in\mathcal{H}'} -\frac{1}{2}\int f(x)^2 d\rho_\theta(x) - \frac{1}{2}\lambda\|f\|_{\mathcal{H}}^2 - \frac{1}{2}\mathcal{U}_\theta(f)^T D(\theta)^{-1}\mathcal{U}_\theta(f) \\
&= -\frac{1}{2}\inf_{f\in\mathcal{H}'}\mathcal{K}(f)
\end{aligned} \tag{20}$$

Since the optimization problem (19) is quadratic, the solution is unique. Therefore, the saddle point problem (17) has a unique solution, and the kernelized Fisher-Rao natural gradient satisfies $\widetilde{\nabla}^F \mathcal{L}(\theta) = \frac{1}{\epsilon} D(\theta)^{-1} \mathcal{U}_\theta(f^*)$, with f^* the unique solution of $\inf_{f \in \mathcal{H}'} \mathcal{K}(f)$. □

Consider now, similarly to Section 3.3, N samples $(Z_n)_{1 \leq n \leq N}$ from the latent distribution ν, which are used to produce N samples $(X_n)_{1 \leq n \leq N}$ from ρ_θ using the map h_θ, i.e., $X_n = h_\theta(Z_n)$. By introducing basis points $(Y_m)_{1 \leq m \leq M}$ uniformly drawn from $(X_n)_{1 \leq n \leq N}$ with $M \leq N$, and M indices $(i_m)_{1 \leq m \leq M}$ uniformly drawn from $[\![1, d]\!]$, we can consider the Nyström subspace $\mathcal{H}_M$:

$$\mathcal{H}_M := span\{x \mapsto \partial_{i_m} k(Y_m, x) \mid 1 \leq m \leq M\}. \tag{21}$$

An estimator for the kernelized Fisher-Rao natural gradient will then be:

$$\widehat{\nabla^F \mathcal{L}(\theta)} = \frac{1}{\epsilon} D(\theta)^{-1} \hat{\mathcal{U}}_\theta(\widehat{f^*}) \tag{22}$$

where $\widehat{f^*}$ is the unique solution to the quadratic optimization problem:

$$\widehat{f^*} := \arg \min_{\substack{f \in \mathcal{H}_M \\ \int f(x) d\rho_\theta(x) = 0}} \frac{1}{N} \sum_{n=1}^{N} f(X_n)^2 + \frac{1}{\epsilon} \hat{\mathcal{U}}_\theta(f)^T D(\theta)^{-1} \hat{\mathcal{U}}_\theta(f)^T + \lambda \|f\|_{\mathcal{H}}^2 \tag{23}$$

Relying on the generalized representer theorem [Schölkopf et al., 2001], we can deduce a closed form expression for the kernelized Fisher-Rao natural gradient, as it is described in the following proposition.

Proposition 6. *The estimator in (22) of the kernelized Fisher-Rao natural gradient satisfies:*

$$\widehat{\nabla^F \mathcal{L}(\theta)} = \frac{1}{\epsilon} \left(D(\theta)^{-1} - D(\theta)^{-1} T^T \left(T D(\theta)^{-1} T^T + \lambda \epsilon K + \frac{\epsilon}{N} A A^T \right)^\dagger T D(\theta)^{-1} \right) \widehat{\nabla \mathcal{L}(\theta)} \tag{24}$$

where A and K are matrices in $\mathbb{R}^{M \times N}$ and $\mathbb{R}^{M \times M}$ given by:

$$A_{m,n} = \partial_{i_m} k(Y_m, X_n), \quad K_{m,m'} = \partial_{i_m} \partial_{i_{m'}+d} k(Y_m, Y_{m'}), \tag{25}$$

while T is a matrix in $\mathbb{R}^{M \times q}$ obtained as the Jacobian of $\theta \mapsto \tau(\theta)$ where:

$$\forall m \in [\![1, M]\!], \quad (\tau(\theta))_m = \frac{1}{N} \sum_{n=1}^{N} \partial_{i_m} k\left(Y_m, h_\theta(Z_n)\right). \tag{26}$$

Proof. Denote $\mathcal{R}(f) = \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta h_\theta(Z_n)^T \nabla f(X_n)$, and B, C matrices in $\mathbb{R}^{Nd \times q}$, $\mathbb{R}^{M \times Nd}$ with:

$$B_{(n,i),j} = (\nabla_\theta h_\theta(Z_n))_{ij}, \qquad C_{m,(n,i)} = \partial_{i_m} \partial_{i+d} k(Y_m, X_n) \tag{27}$$

where $\nabla_\theta h_\theta(Z_n)$ the Jacobian matrix of $\theta \mapsto h_\theta(Z_n)$ for a fixed Z_n. Let $f \in \mathcal{H}_M$, and decompose it as $f = \sum_{m=1}^{M} \alpha_m \partial_{i_m} k(Y_m, \cdot)$. Using the reproducing property $\partial_i f(x) = \langle f, \partial_i k(x, \cdot) \rangle_{\mathcal{H}}$ [Steinwart and Christmann, 2008], we have:

$$\begin{aligned} \|f\|_{\mathcal{H}}^2 &= \langle \sum_{m=1}^{M} \alpha_m \partial_{i_m} k(Y_m, \cdot), \sum_{m=1}^{M} \alpha_m \partial_{i_m} k(Y_m, \cdot) \rangle_{\mathcal{H}} \\ &= \sum_{1 \leq m, m' \leq M} \alpha_m \alpha_{m'} \langle \partial_{i_m} k(Y_m, \cdot), \partial_{i_{m'}} k(Y_{m'}, \cdot) \rangle_{\mathcal{H}} \\ &= \sum_{1 \leq m, m' \leq M} \alpha_m \alpha_{m'} \partial_{i_m} \partial_{i_{m'}+d} k(Y_m, Y_{m'}) \\ &= \alpha^T K \alpha \end{aligned} \tag{28}$$

Denoting the i-th row of $\nabla_\theta h_\theta(Z_n)$ by $(\nabla_\theta h_\theta(Z_n))_i \in \mathbb{R}^q$ for $i \in [\![1, d]\!]$, we also have:

$$\begin{aligned}
\mathcal{R}(f) &= \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{d} \partial_i f(X_n)(\nabla_\theta h_\theta(Z_n))_i \\
&= \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{d} \langle f, \partial_i k(X_n, \cdot)\rangle_{\mathcal{H}}(\nabla_\theta h_\theta(Z_n))_i \\
&= \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{d} \langle \sum_{m=1}^{M} \alpha_m \partial_{i_m} k(Y_m, \cdot), \partial_i k(X_n, \cdot)\rangle_{\mathcal{H}}(\nabla_\theta h_\theta(Z_n))_i \\
&= \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{d}\sum_{m=1}^{M} \alpha_m \partial_{i_m}\partial_{i+d} k(Y_m, X_n)(\nabla_\theta h_\theta(Z_n))_i \\
&= \frac{1}{N}(CB)^T\alpha \\
&= T^T\alpha
\end{aligned} \tag{29}$$

where $\frac{1}{N}CB = T$ because of the chain rule.

Finally, we have:

$$\begin{aligned}
\frac{1}{N}\sum_{n=1}^{N} f(X_n) &= \frac{1}{N}\sum_{n=1}^{N}\sum_{m=1}^{M} \alpha_m \partial_{i_m} k(Y_m, X_n) \\
&= \frac{1}{N}\alpha^T A A^T \alpha.
\end{aligned} \tag{30}$$

Therefore, the optimal solution $\widehat{f^*}$ in (23) is given by $\widehat{f^*} = \sum_{m=1}^{M} \alpha_m^* \partial_{i_m} k(Y_m, \cdot)$ where α^* is the solution of the problem:

$$\min_{\alpha\in\mathbb{R}^M} \alpha^T\left(\frac{\epsilon}{N}AA^T + \epsilon\lambda K + TD(\theta)^{-1}T^T\right)\alpha + 2\alpha TD(\theta)^{-1}\widehat{\nabla\mathcal{L}(\theta)}. \tag{31}$$

The solution of this problem is:

$$\alpha^* = -\left(\frac{\epsilon}{N}AA^T + \epsilon\lambda K + TD(\theta)^{-1}T^T\right)^{\dagger} TD(\theta)^{-1}\widehat{\nabla\mathcal{L}(\theta)}. \tag{32}$$

Finally, we conclude by remarking that the kernelized Fisher-Rao natural gradient from expression (22) can be written as $\widehat{\nabla^F\mathcal{L}(\theta)} = \frac{1}{\epsilon}D(\theta)^{-1}\hat{\mathcal{U}}_\theta(\widehat{f^*}) = \frac{1}{\epsilon}D(\theta)^{-1}\left(\widehat{\nabla\mathcal{L}(\theta)} + T^T\alpha^*\right)$.

□

We will study the performances of the derived estimator of Fisher-Rao natural gradient in the Section 5.

5 Experiments

In this section we reproduce some experiments proposed in [Arbel et al., 2019] to illustrate the performance of the estimator of KWNG derived from Section 3.

5.1 Experiments context

Parametric model In the following experiments, we consider a multivariate normal model with diagonal matrix in dimension d, called $\mathcal{P}_\Theta = \{\rho_\theta = (h_\theta)_\# \nu \parallel \theta \in \Theta\}$. In this context, the sample space is $\Omega = \mathbb{R}^d$, and the parameter space is $\Theta = \mathbb{R}^d \times \mathbb{R}_+^d$. The latent distribution is $\nu \sim \mathcal{N}(0, Id)$ a standard normal distribution in dimension d. And for all parameters $\theta = (\mu, \delta) \in \mathbb{R}^d \times \mathbb{R}_+^d$, h_θ is the function $z \mapsto diag\,(\delta)^{\frac{1}{2}} z + \mu$, where $diag : \mathbb{R}^d \mapsto \mathbb{R}^{d\times d}$ is the operator mapping a vector to its corresponding diagonal matrix. The diagonal matrix of $(\delta_i)_i \in \mathbb{R}^d$ is denoted Δ.

For all $z = (z_i)_i \in \mathbb{R}^d$, the Jacobian matrix of the function $\theta \mapsto h_\theta(z)$ is:

$$\forall \theta \in \Theta, \quad \forall (k,l) \in [\![1,d]\!] \times [\![1,2d]\!], \quad (\nabla_\theta h_\theta(z))_{kl} = \begin{cases} 1, & \text{if } k = l \\ \frac{z_i}{2\sqrt{\delta_i}}, & \text{if } l = k + d \\ 0, & \text{otherwise.} \end{cases} \tag{33}$$

Loss function Consider a target parameter $\theta^* \in \times$. In the following experiments, it will be the parameter corresponding to the standard normal distribution. The loss function $\mathcal{L} : \theta \mapsto \mathcal{L}(\theta)$ defined over Θ is the Wasserstein distance squared between the distribution ρ_θ of the parametric model $\mathcal{P}_\Theta$, and the target distribution ρ_{θ^*}. We have:

$$\begin{aligned} \forall \theta \in \Theta, \quad \mathcal{L}(\theta) &= W_2^2(\rho_\theta, \rho_{\theta^*}) \\ &= \|\mu - \mu^*\|_2^2 + \frac{1}{2} tr\left(\Delta + \Delta^* - 2\sqrt{\Delta^{\frac{1}{2}} \Delta^* \Delta^{\frac{1}{2}}}\right) \\ &= \|\mu - \mu^*\|_2^2 + \sum_{i=1}^d \left(\sqrt{\delta_i} - \sqrt{\delta_i^*}\right)^2 \end{aligned} \tag{34}$$

since we consider Gaussian distributions, and the covariance matrices are diagonal. The gradient of the loss function is then:

$$\forall \theta \in \Theta, \quad \nabla \mathcal{L}(\theta) = (\nabla_\mu \mathcal{L}(\theta), \nabla_\delta \mathcal{L}(\theta)) = \left(2(\mu - \mu^*), \left(\frac{\sqrt{\delta_i} - \sqrt{\delta_i^*}}{\sqrt{\delta_i}}\right)_{i \in [\![1,d]\!]}\right). \tag{35}$$

In the following experiments, when we need to compute an estimator of the gradient $\widehat{\nabla \mathcal{L}(\theta)}$, we will instead directly use the gradient $\nabla \mathcal{L}(\theta)$.

Exact Wasserstein natural gradient The appendix D.1 in [Arbel et al., 2019] gives the exact Wasserstein natural gradient for a multivariate normal model. We can use this result to compute the exact Wasserstein natural gradient (WNG) in our model:

$$\nabla_\theta^W \mathcal{L}(\theta) = \left(\nabla_\mu \mathcal{L}(\theta), diag^{-1}\left(\Delta(A + diag(A)) + (A + diag(A))\Delta\right)\right) \tag{36}$$

where $A = diag^{-1}\left(\nabla_\delta \mathcal{L}_\theta(\theta)\right)$ and $diag^{-1}$ is the inverse operator of $diag$. Using derivations from (35), the exact Wasserstein natural gradient in our model:

$$\nabla_\theta^W \mathcal{L}(\theta) = \left(2(\mu - \mu^*), \left(4\left(\delta_i - \sqrt{\delta_i \delta_i^*}\right)\right)_{i \in [\![1,d]\!]}\right) \tag{37}$$

The considered model is chosen such that we can compute the exact Wasserstein natural gradient. In the experiments, we will compare the estimator of KWNG with this exact Wasserstein natural gradient.

Exact Fisher-Rao natural gradient The Fisher information matrix for a multivariate normal model is given in [Malagò and Pistone, 2015]. In our model with diagonal covariance matrix, the Fisher information matrix is:

$$G^F(\theta) = diag\left(\frac{1}{\delta_1}, \cdots, \frac{1}{\delta_d}, \frac{1}{2\delta_1^2}, \cdots, \frac{1}{2\delta_d^2}\right) \tag{38}$$

so that the exact Fisher-Rao natural gradient is given by:

$$\nabla^F \mathcal{L}(\theta) = G^F(\theta)^{-1} \nabla \mathcal{L}(\theta). \tag{39}$$

5.2 Implementation of the kernelized natural gradient estimator

We implemented the estimator of KWNG in Proposition 3 for our considered model and using one of the kernels described in Section 5.3 in Python for testing its performances and its properties. We also implemented the estimator of KFRNG derived in Proposition 6. After trying several choices of kernels and parameters, we didn't manage to get a consistent estimator for the kernelized Fisher-Rao natural gradient, as it is shown in Figure 1. We compare the relative error between the estimator of KFRNG $\widehat{\nabla^F \mathcal{L}(\theta)}$ and the exact Fisher-Rao natural gradient $\nabla^F \mathcal{L}(\theta)$:

$$e^F(\theta) := \frac{\|\widehat{\nabla^F \mathcal{L}(\theta)} - \nabla^F \mathcal{L}(\theta)\|}{\|\nabla^F \mathcal{L}(\theta)\|} \tag{40}$$

for a given number of random parameters θ, and for different choices of parameters σ_0 of the Gaussian kernel (see section 5.3). The estimated KFRNG is completely different from the exact Fisher-Rao gradient: the estimated gradient $\widehat{\nabla^F \mathcal{L}(\theta)}$ is too small compared to the values of the exact gradient $\nabla^F \mathcal{L}(\theta)$ for small values of σ_0, and too large for large values of σ_0. In the following experiments, we will focus only on the implementation of the estimator of KWNG. We leave the test of the estimator of KFRNG derived in Proposition 6 as a future work.

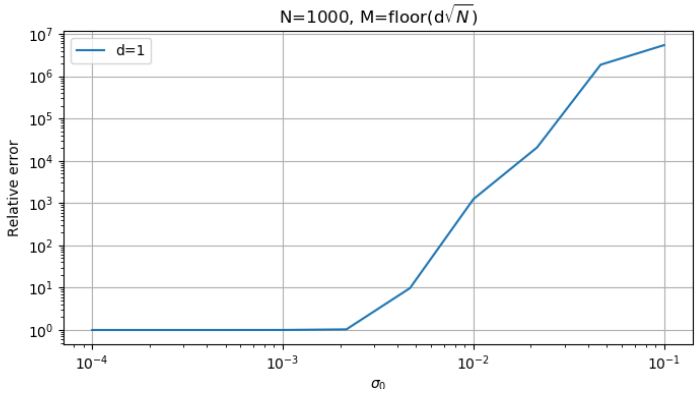

Figure 1: Relative error between the estimated KFRNG in Proposition 6 and the exact Fisher-Rao natural gradient, with respect to several values of parameter σ_0 in the choice of Gaussian kernel (see Section 5.3). The relative error in the graph is the mean error computed from 10 different random parameters. Experience parameters: $\epsilon = 10^{-10}$, $\lambda = 10^{-10}$, $N = 1000$, $M = \lfloor d\sqrt{N} \rfloor$, $d = 1$.

The implementation of the KWNG estimator works on our model, for small dimensions as we will see in the following experiments. The computation time of our implementation is summarized in Figure 2. The computational cost increases with M the number of basis points and the dimension d, as expected by the theory in Section 3.4. With our implementation, we will limit our experiments to problems with dimension smaller than 5, because the computational cost is to heaavy for higher dimensions with our implementation. The sample size will be chosen between 10^2 and 10^3, and the number of basis points $M \sim d\sqrt{N}$ is enough for a good trade-off between accuracy and computational cost. The sample size $N = 10^3$ is too computationally heavy to use the estimator in a steepest descent method. In practice, we will prefer samples of size $N = 10^2$.

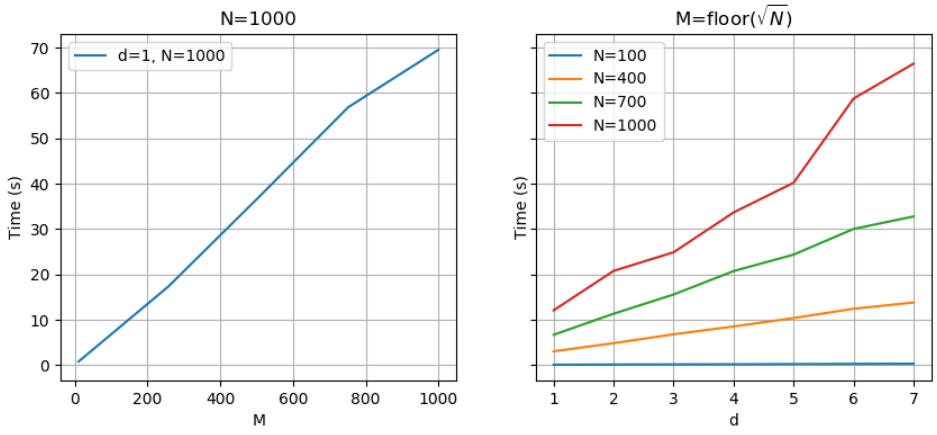

Figure 2: Computational cost of the estimator of KWNG, with respect to the number of basis points M and the dimension of the problem d. Left: we fix the number of samples $N = 1000$, and measure the time needed to compute an estimator for a given number of basis points M in our implementation. Right: we choose several values for the number of samples N, and the number of basis points is $M = \left\lfloor d\sqrt{N} \right\rfloor$; we then measure the time to compute the estimator of KWNG in problems of different dimension d. Experience parameters: Gaussian kernel with $\sigma_0 = 5$ (see Section 5.3), $\epsilon = 10^{-10}$, $\lambda = 10^{-10}$.

5.3 Choice of kernel

In this section, we compare in practice the Gaussian kernel and the sigmoid kernel used to construct the estimator of KWNG, which are kernels satisfying the conditions in Section 3.5.

Definition 4 (Gaussian kernel)**.** *For a parameter $\sigma > 0$, the Gaussian kernel in Ω is defined as:*

$$\forall a, b \in \Omega, \quad k(a,b) = \exp\left(-\frac{\|a-b\|^2}{2\sigma}\right). \tag{41}$$

The partial derivatives of the Gaussian kernel are, for all $i, j \in [\![1, d]\!]$:

$$\begin{aligned} \forall a, b \in \Omega, \quad \partial_i k(a,b) &= -\frac{1}{\sigma^2}(a_i - b_i)\exp\left(-\frac{\|a-b\|^2}{2\sigma}\right) \\ \partial_i \partial_{j+d} k(a,b) &= \frac{1}{\sigma^2}\exp\left(-\frac{\|a-b\|^2}{2\sigma}\right)\left(-\frac{1}{\sigma^2}(a_i - b_i)(a_j - b_j) + \mathbb{1}\{i=j\}\right) \end{aligned} \tag{42}$$

where we recall that the notation $\partial_i k(a,b)$ is the partial derivative of $a \mapsto k(a,b)$ with respect to a_i, the i-th coordinate of $a \in \Omega$, when $b \in \Omega$ is fixed, and the notation $\partial_{j+d} k(a,b)$ is the partial derivative of $b \mapsto k(a,b)$ with respect to b_j.

Definition 5 (Sigmoid kernel)**.** *For parameters $\alpha, c \in \mathbb{R}$, the sigmoid kernel in Ω is defined as:*

$$\forall a, b \in \Omega, \quad k(a,b) = \tanh(\alpha a^T b + c) \tag{43}$$

The partial derivatives of the sigmoid kernels are, for all $i, j \in [\![1, d]\!]$:

$$\begin{aligned} \forall a, b \in \Omega, \quad \partial_i k(a,b) &= (1 - k(a,b)^2)(\alpha b_i + c) \\ \partial_i \partial_{i+d} k(a,b) &= (1 - k(a,b)^2)\left(\alpha \mathbb{1}\{i=j\} - 2k(a,b)(\alpha a_j + c)(\alpha b_i + c)\right). \end{aligned} \tag{44}$$

Kernel in practice In practice, the choice of the kernel and its parameter is important to get an estimator of KWNG converging to the exact Wasserstein natural gradient. The parameter of the kernel should be adapted to the samples $(X_n)_{1\leq n\leq N}$ and $(Y_m)_{1\leq m\leq M}$:

- the parameter σ of the Gaussian kernel is $\sigma = \sigma_0 \sigma_{N,M}$ where $\sigma_0 > 0$ is a scale factor that we have to fine-tune and $\sigma_{N,M}$ is the mean of the squared Euclidean distance between the samples X_n and the basis points Y_m for $1 \leq n \leq N$, $1 \leq m \leq M$;
- the parameters α and c of the sigmoid kernel are $\alpha = s_0 \frac{1}{\alpha_{N,M}}$ and $c = -s_0 \frac{c_{N,M}}{\alpha_{N,M}}$ where $c_{N,M}$ and $\alpha_{N,M}$ are the mean and the standard deviation of all the scalar products between X_n and Y_m for $1 \leq n \leq N$, $1 \leq m \leq M$, and s_0 is a scale factor which has to be fine-tuned.

In Figure 3 and 4, we fine-tune the parameter by comparing the relative errors between the estimated KWNG $\widehat{\nabla^W \mathcal{L}(\theta)}$ and the exact Wasserstein natural gradient $\nabla^W \mathcal{L}(\theta)$:

$$e^W(\theta) := \frac{\|\widehat{\nabla^W \mathcal{L}(\theta)} - \nabla^W \mathcal{L}(\theta)\|}{\|\nabla^W \mathcal{L}(\theta)\|} \tag{45}$$

at different values of the kernel parameter (σ_0 for the Gaussian kernel, s_0 for the sigmoid kernel), and for a given number of random parameters θ. The Gaussian and the sigmoid kernels both have values of parameters σ_0 and s_0 for which the relative error is acceptable, as long as the dimension isn't too large. The estimation is harder when using the sigmoid kernel, since it is very sensitive to the scale parameter s_0. For $s_0 \geq 10^{-15}$, the relative error is 1, because the norm of the estimated KWNG $\|\widehat{\nabla^W \mathcal{L}(\theta)}\|$ is very small compared to the norm of the exact Wasserstein natural gradient $\|\nabla^W \mathcal{L}(\theta)\|$, while for $s_0 \leq 10^{-15}$, $\|\widehat{\nabla^W \mathcal{L}(\theta)}\|$ is very large compared to $\|\nabla^W \mathcal{L}(\theta)\|$. Moreover, with the sigmoid kernel, while the estimation is relatively good in dimension $d = 1$, it becomes very bad with $d \geq 2$ compared to the estimation with Gaussian kernel. In the following experiments, we will prefer using the Gaussian kernel for the estimator of KWNG.

5.4 Consistency of the estimator

In these experiments, we reproduce the Figure 1 in [Arbel et al., 2019]. After implementing the estimator of KWNG according to Proposition 3, we study its behavior in the limit of large N and M, for two different choices of kernel, the Gaussian kernel and the sigmoid kernel, with fine-tuned parameters as described in Section 5.3.

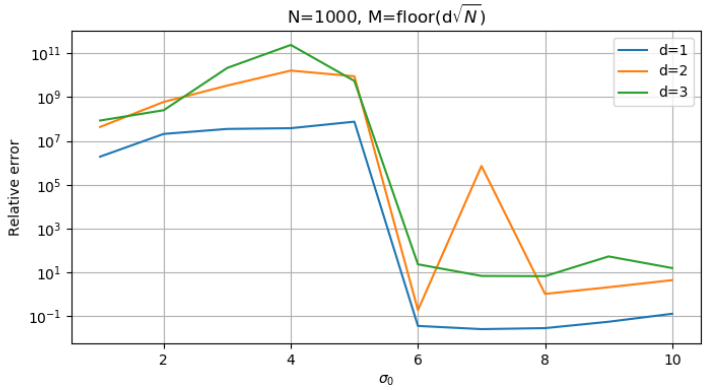

Figure 3: Fine-tuning of the parameter σ_0 of the Gaussian kernel: relative error between the estimator of KWNG in Proposition 3 and the exact Wasserstein natural gradient, with respect to the parameter σ_0 of the Gaussian kernel. We represent the relative error averaged over 10 different random values of the parameter θ. Experience parameters: $\epsilon = 10^{-10}$, $\lambda = 10^{-10}$, $N = 1000$, $M = \lfloor d\sqrt{N} \rfloor$, $d = 1, 2, 3$.

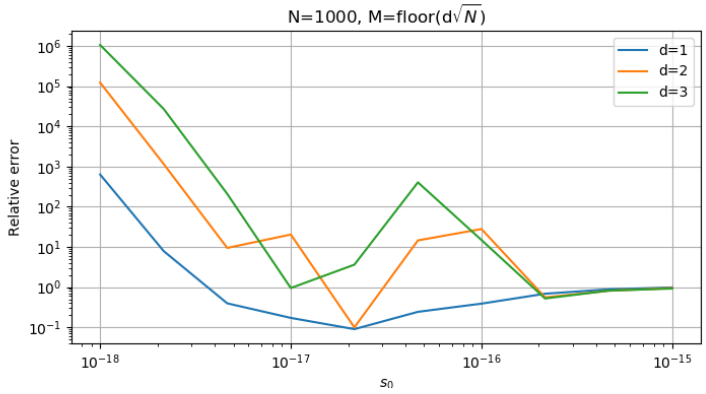

Figure 4: Fine-tuning of the parameter s_0 of the sigmoid kernel: relative error between the estimator of KWNG in Proposition 3 and the exact Wasserstein natural gradient, with respect to the parameter s_0 of the sigmoid kernel. We represent the relative error averaged over 10 different random values of the parameter θ. Experience parameters: $\epsilon = 10^{-10}$, $\lambda = 10^{-10}$, $N = 1000$, $M = \lfloor d\sqrt{N} \rfloor$, $d = 1, 2, 3$.

For a given dimension d of the problem, and a given number of samples N and M, we compute the estimator of KWNG $\widehat{\nabla^W \mathcal{L}(\theta)}$ and compare it to the exact value of the WNG $\nabla^W \mathcal{L}(\theta)$ given by (37), for several random parameters $\theta \in \Theta$. The comparison is done by computing the relative error as described in (45).

As the dimension of the problem gets higher, the estimation problem is harder, as we can see in the Figure 7. This is confirmed by the theory in Section 3.5 where the computational cost increases with the dimension d and the number of parameters $q = 2d$ in our experiments. The relative error is too large for dimensions higher than 5, and it wouldn't make any sense to use it in practice as the steepest descent direction. We also observe the convergence of the estimator of KWNG in large N and M, as shown in Figures 5 and 6. However, the convergence gets more and more difficult as the dimension of the problem gets higher. These experiments show that it is not necessary to choose a sample size very large to get a good accuracy for the estimator. In practice, $N = 10^2$ is enough, with $M = \lfloor d\sqrt{N} \rfloor$. The next step is the evaluate to performance of the estimator of KWNG in a steepest descent method.

5.5 Performance in steepest descent method

In this experience, given a target parameter θ^* which corresponds to the standard normal distribution, we implement the steepest descent method to evaluate the quality of the estimator of KWNG in an optimization problem. Recall that the

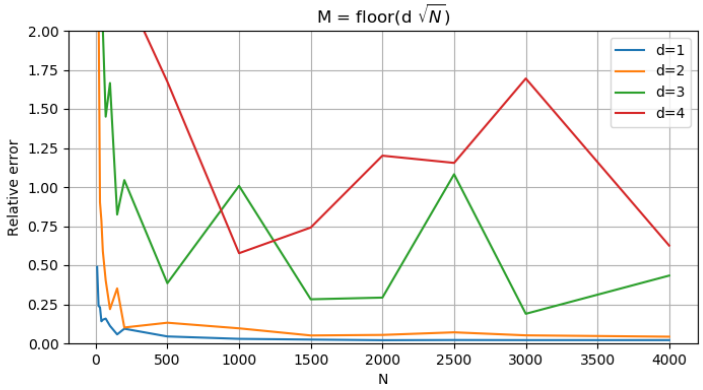

Figure 5: Consistency of the estimator of KWNG, with respect to the number of samples N. For each values of N, we fix the number of basis points $M = \lfloor d\sqrt{N} \rfloor$, and for dimension $d = 1, 2, 3, 4$, we compute the mean relative errors over 10 random parameters θ. Experience parameters: Gaussian kernel with parameter $\sigma_0 = 5, \epsilon = 10^{-10}$, $\lambda = 10^{-10}$.

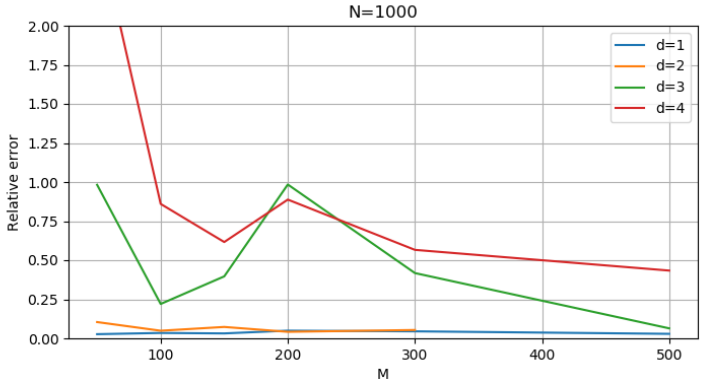

Figure 6: Consistency of the estimator of KWNG, with respect to the number of basis points M. Each relative error represented is the mean error over 10 random parameters θ. We compare the performances of the estimator for several dimensions $d = 1, 2, 3, 4$. Experience parameters: Gaussian kernel with parameter $\sigma_0 = 5, \epsilon = 10^{-10}$, $\lambda = 10^{-10}$, $N = 1000$.

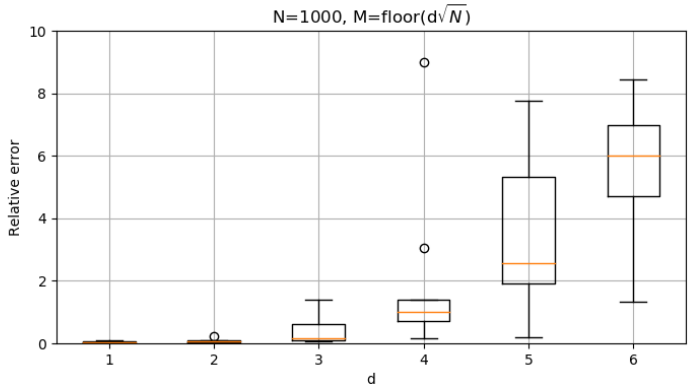

Figure 7: Consistency of the estimator of KWNG, for different dimensions d of the problem. For each dimension d, we compute 10 estimations for different random parameters θ, and plot the relative errors with box plot. Experience parameters: Gaussian kernel with parameter $\sigma_0 = 5$, $\epsilon = 10^{-10}$, $\lambda = 10^{-10}$, $N = 1000$, $M = \lfloor d\sqrt{N} \rfloor$.

steepest descent method corresponds to the following update rule:

$$\theta_{t+1} = \theta_t - \alpha_{t+1}\widetilde{\nabla}\mathcal{L}(\theta_t) \tag{46}$$

where $\widetilde{\nabla}\mathcal{L}(\theta)$ denotes the steepest direction and $(\alpha_t)_t$ are the step sizes. In the following experiments, we will choose the following steepest directions:

1. the Euclidean gradient $\nabla\mathcal{L}(\theta)$;
2. the exact Wasserstein natural gradient $\nabla^W\mathcal{L}(\theta)$;
3. the estimator of KWNG $\widehat{\nabla^W\mathcal{L}(\theta)}$ from Proposition 3;
4. the exact Fisher-Rao natural gradient $\nabla^F\mathcal{L}(\theta)$.

For comparison purpose, the step sizes will be the same for all these descent methods. The trajectory of the parameters from the initial parameter to the target parameters is visualized in the two-dimensional plane obtained by the PCA decomposition of the parameters sequence obtained in the fastest steepest descent method, which is the one using the exact Wasserstein natural gradient. Figure 8 shows the comparison how the different descent direction.

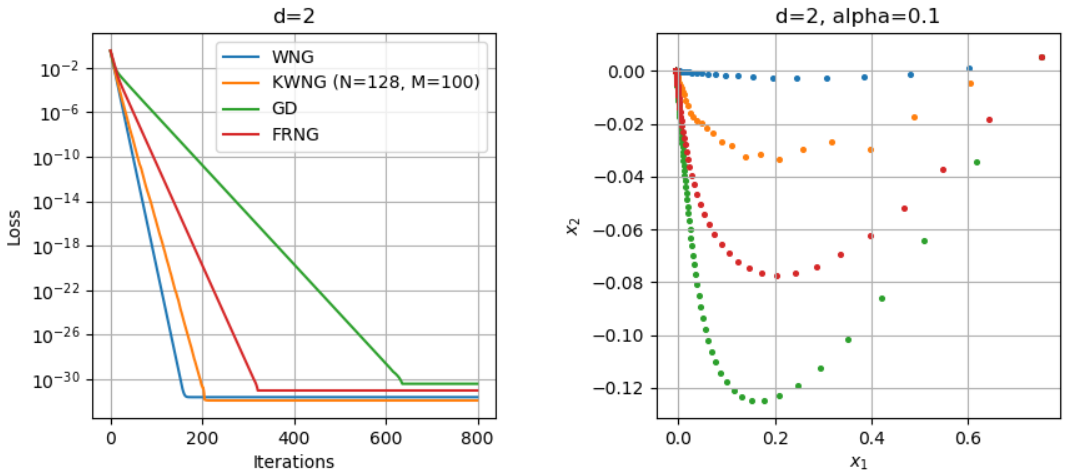

Figure 8: Comparison of steepest descent methods with different type of gradients: exact Wasserstein natural gradient (WNG), estimator of KWNG in Proposition 3 (KWNG), Euclidean gradient (GD) and exact Fisher-Rao natural gradient (FRNG). Left: loss function over the iterations of the descent method. Right: trajectory of the parameters for each of the previous descent methods, projected on the two-dimensional plane obtained after PCA decomposition of the parameters in the descent method using exact Wasserstein natural gradient; x_1 is the first eigenvector in the PCA decompostion, while x_2 is the second eigenvector; the initial parameter is $\theta_0 = (0.7, 0.05)$ and the target parameter is $\theta^* = (0, 0)$. Experience parameters: dimension of the problem $d = 2$, fixed step size for all methods $\alpha_t = 0.1$, Gaussian kernel with $\sigma = 5$, $N = 128$, $M = 100$, $\epsilon = 10^{-10}$, $\lambda = 10^{-10}$ for the estimation of KWNG.

This experience illustrates the efficiency of the Wasserstein natural gradient, compared in particular to the gradient descent method using Euclidean gradient which is sensible to the curvature of the problem. The trajectory of the parameters from the initial parameter θ_0 to the target parameter θ^* is more direct when using the Wasserstein natural gradient. The estimator of KWNG is accurate enough in practice so that the parameter trajectory of a steepest descent method using this estimation of the KWNG is a good approximation of the trajectory with exact Wasserstein natural gradient. It outperforms the basic gradient descent in this task.

We also observe that Wasserstein natural gradient is a more efficient steepest direction than the Fisher-Rao natural gradient. This results has been also shown in the experiences of [Arbel et al., 2019], when Figure 3 and 4 compare the performance of steepest descent methods using estimation of KWNG, KFAC and EKFAC on a deep neural network optimization tasks. In these experiences, methods using estimation of KWNG performs better than the ones using KFAC and EKFAC which are approximations of the Fisher-Rao natural gradient. However, it is not clear theoretically why the Wasserstein natural gradient is better than the Fisher-Rao natural gradient in general.

This experience a general illustration on how the natural gradient can efficiently use the curvature of the manifold of the parametric model for the optimization process, because the Euclidean gradient descent is slower compared to natural gradient during the optimization.

6 Conclusion and perspective

This project studied [Arbel et al., 2019] which presented a method based on kernel estimators to estimate efficiently the Wasserstein natural gradient. It is shown in the experiments of the paper that the proposed estimator of KWNG can be used efficiently for natural gradient descent methods, for example in deep neural networks. The main advantage of using natural gradient is its robustness to ill-conditioning of the problem.

After reproducing an implementation of this method, we tested in this project the consistency of the estimator and its efficiency in a steepest descent method. We indeed found that steepest descent method with the estimated KWNG outperforms method using Euclidean gradient or Fisher-Rao natural gradient, as long as the estimation is consistent. However, results of our experiments on the multivariate normal model showed that the estimator doesn't converge well when the problem gets difficult in high dimension, which makes the steepest descent method inoperable in these cases. The estimation method is also very sensible to the choice of the kernel and its parameter. A theory about practical choice of the kernel for the estimation of KWNG with this method should be elaborated.

The current method can be improved by making it less sensitive to the choice of the kernel, by using some theoretical properties based on the works about kernel methods. It also needs more robustness when dealing with high dimensional problems. Another improvement is to find other approximations in addition to the use of kernel estimators, so that we can compute a faster estimator of the natural gradient.

Possible future works include a precise comparison between Fisher-Rao natural gradient and the Wasserstein natural gradient, to understand why the second one performs better than the first one. It will be particularly interesting to investigate the properties and the experimental performances of the estimator of the Fisher-Rao natural gradient derived in this project based on the method proposed by [Arbel et al., 2019].

References

[Amari, 1985] Amari, S. (1985). Differential-geometrical methods in statistics.

[Amari, 1998] Amari, S. (1998). Natural gradient works efficiently in learning. *Neural Computation*, 10:251–276.

[Arbel et al., 2019] Arbel, M., Gretton, A., Li, W., and Montúfar, G. (2019). Kernelized wasserstein natural gradient. *ArXiv*, abs/1910.09652.

[Boyd and Vandenberghe, 2004] Boyd, S. P. and Vandenberghe, L. (2004). Convex optimization. *IEEE Transactions on Automatic Control*, 51:1859–1859.

[Chen and Li, 2018] Chen, Y. and Li, W. (2018). Natural gradient in wasserstein statistical manifold. *ArXiv*, abs/1805.08380.

[Duchi et al., 2010] Duchi, J. C., Hazan, E., and Singer, Y. (2010). Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159.

[Ekeland and Téman, 1976] Ekeland, I. and Téman, R. (1976). Convex analysis and variational problems.

[George et al., 2018] George, T., Laurent, C., Bouthillier, X., Ballas, N., and Vincent, P. (2018). Fast approximate natural gradient descent in a kronecker-factored eigenbasis. In *NeurIPS*.

[Hinton et al., 2012] Hinton, G., Srivastava, N., and Swersky, K. (2012). Lecture 6a overview of mini–batch gradient descent.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

[Li et al., 2019] Li, W., Lin, A. T., and Montúfar, G. (2019). Affine natural proximal learning. In *GSI*.

[Malagò and Pistone, 2015] Malagò, L. and Pistone, G. (2015). Information geometry of the gaussian distribution in view of stochastic optimization. In *FOGA*.

[Martens and Grosse, 2015] Martens, J. and Grosse, R. B. (2015). Optimizing neural networks with kronecker-factored approximate curvature. *ArXiv*, abs/1503.05671.

[Schölkopf et al., 2001] Schölkopf, B., Herbrich, R., and Smola, A. J. (2001). A generalized representer theorem. In *COLT/EuroCOLT*.

[Steinwart and Christmann, 2008] Steinwart, I. and Christmann, A. (2008). Support vector machines. In *Information science and statistics*.