Fisher Information and Natural Gradient Learning in Random Deep Networks

Shun-ichi Amari
RIKEN CBS, Wako-shi, Japan

Ryo Karakida
AIST, Tokyo, Japan

Masafumi Oizumi
Araya Inc., Tokyo, Japan

Abstract

The parameter space of a deep neural network is a Riemannian manifold, where the metric is defined by the Fisher information matrix. The natural gradient method uses the steepest descent direction in a Riemannian manifold, but it requires inversion of the Fisher matrix, however, which is practically difficult. The present paper uses statistical neurodynamical method to reveal the properties of the Fisher information matrix in a net of random connections. We prove that the Fisher information matrix is unit-wise block diagonal supplemented by small order terms of off-block-diagonal elements. We further prove that the Fisher information matrix of a single unit has a simple reduced form, a sum of a diagonal matrix and a rank 2 matrix of weight-bias correlations. We obtain the inverse of Fisher information explicitly. We then have an explicit form of the approximate natural gradient, without relying on the matrix inversion.

1 INTRODUCTION

In modern deep learning, multilayer neural networks are usually trained by using the stochastic gradient-descent method (See Amari, 1967 for one of the earliest proposal of stochastic gradient descent for the purpose of applying it to multilayer networks). The parameter space of multilayer networks forms a Riemannian space equipped with Fisher information metric. Thus, instead of the usual gradient descent method, the natural gradient or Riemannian gradient method, which takes account of the geometric structure of the Riemmanian space, is more effective for learning (Amari, 1998). However, it has been difficult to apply the natural gradient descent because it needs the inversion of the Fisher information matrix, which is computationally heavy. Many approximation methods reducing computational costs have therefore been proposed (see Pascanu & Bengio, 2013; Ollivier, 2015; Grosse & Martens, 2016; Martens, 2017).

To resolve the computational difficulty of the natural gradient, we use a neural network of random connections and analyze its Fisher information matrix. We prove that, when the number n of neural units in each layer is sufficiently large, the subblocks of the Fisher information matrix $\boldsymbol{G}$ corresponding to different layers are of order $1/\sqrt{n}$, which is negligibly small. Thus, $\boldsymbol{G}$ is approximated by a layer-wise diagonalized matrix. Furthermore, within the same layer, the subblocks among different units are also of order $1/\sqrt{n}$. This justifies the use of unit-wise diagonalized $\boldsymbol{G}$.

This gives a justification for the unit-wise natural gradient method proposed by Kurita (1994) and studied in detail by Ollivier (2015) and Marceau-Caron and Ollivier (2016). We further study the Fisher information matrix of a unit —that is, a simple perceptron— for the purpose of implementing unit-wise natural gradient learning. We obtain an explicit form of the Fisher information matrix and its inverse under the assumption that inputs are subject to the standard non-correlated Gaussian distribution with mean 0. This assumption is approximately satisfied when we use a residual network with ReLU activation functions. The unit-wise natural gradient is explicitly formulated without matrix inversion, making it possible that natural gradient learning is realized without the burden of heavy computation. Although, our approximation method is justified only for random networks under the mean-field assumption, it is expected that it would be effective for training actual deep networks, considering the good performances shown in Ollivier, 2015 and Marceau-Caron and Ollivier, 2016.

The present paper is purely theoretical. It is needless to say that computer simulations are necessary to show the usefulness.

Proceedings of the 22[nd] International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

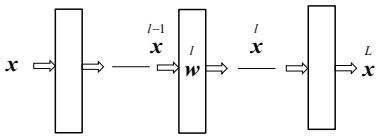

Figure 1: Deep neural network

2 DEEP NEURAL NETWORKS

We consider a deep neural network consisting of L layers. Let $\overset{l-1}{\boldsymbol{x}}$ be the input vectors to the l-th layer and $\overset{l}{\boldsymbol{x}}$ the output vector of the l-th layer (see Figure 1). The input-output relation of the l-th layer is written as

$$\overset{l}{x}_i = \varphi\left(\sum_j \overset{l}{w}_{ij}\,\overset{l-1}{x}_j + \overset{l}{b}_i\right), \tag{1}$$

where φ is an activation function such as a rectified linear function (ReLU), sigmoid function, etc. Let n_l be the number of neurons in the l-th layer. We assume that $n_1, n_2, \cdots, n_{L-1}$ are large, but the number of neurons in the final layer, n_L, can be small. Even $n_L = 1$ is allowed. The weights $\overset{l}{w}_{ij}$ and biases $\overset{l}{b}_j$ are random variables subject to independent Gaussian distributions with mean 0 and variances σ_l^2/n_{l-1} and σ_{bl}^2, respectively. Note that each weight is a random variable of order $1/\sqrt{n_{l-1}}$, but the weighted sum $\sum \overset{l}{w}_{ij}\,\overset{l-1}{x}_j$ is of order 1.

We recapturate briefly the feedforward analysis of input signals given in Poole et al., 2016 and Amari, Karakida and Oizumi, 2018, to introduce the activity $\overset{l}{A}$ and enlargement factor $\overset{l}{\chi}$. They also play a role in the feedback analysis obtaining the Fisher information (Schoenholtz et al., 2016; Karakida, Akaho and Amari, 2018).

Let us put

$$\overset{l}{u}_i = \sum_j \overset{l}{w}_{ij}\,\overset{l-1}{x}_j + \overset{l}{b}_i\,. \tag{2}$$

Given $\overset{l-1}{\boldsymbol{x}}$, $\overset{l}{u}_i$ are independently and identically distributed (iid) Gaussian random variables with mean 0 and variance

$$\tau_l^2 = \frac{\sigma_l^2}{n}\sum\left(\overset{l-1}{x}_j\right)^2 + \sigma_{bl}^2 = \overset{l-1}{A}\,\sigma_l^2 + \sigma_{bl}^2, \tag{3}$$

where

$$\overset{l-1}{A} = \frac{1}{n_{l-1}}\sum \overset{l-1}{x}_j{}^2 \tag{4}$$

is the total activity of input $\overset{l-1}{\boldsymbol{x}}$.

It is easy to show how $\overset{l}{A}$ develops across the layers. Since $x_j^2 = \varphi\left(u_j\right)^2$ are iid when $\overset{l-1}{\boldsymbol{x}}$ is fixed, the law of large numbers guarantees that their sum is replaced by the expectation when n_{l-1} is large. Putting $u_j = \tau_l v$ where v is the standard Gaussian variables, we have a recursive equation,

$$\overset{l}{A} = \int \left\{\varphi\left(\tau_l\, v\right)\right\}^2 Dv, \tag{5}$$

where τ_l in equation (3) depends on $\overset{l-1}{A}$ and

$$Dv = \frac{1}{\sqrt{2\pi}}\exp\left\{-\frac{v^2}{2}\right\}dv. \tag{6}$$

Since equation (1) gives the transformation from $\overset{l-1}{\boldsymbol{x}}$ to $\overset{l}{\boldsymbol{x}}$, we study how a small difference $d\,\overset{l-1}{\boldsymbol{x}}$ in the input develops to give difference $d\,\overset{l}{\boldsymbol{x}}$ in the output. By differentiating equation (1), we have

$$d\,\overset{l}{\boldsymbol{x}} = \overset{l}{\boldsymbol{B}}\,d\,\overset{l-1}{\boldsymbol{x}} \tag{7}$$

where

$$\overset{l}{\boldsymbol{B}} = \frac{\partial\,\overset{l}{\boldsymbol{x}}}{\partial\,\overset{l-1}{\boldsymbol{x}}} \tag{8}$$

is the Jacobian matrix whose (i_l, i_{i-1})-th element is given by

$$B_{i_{l-1}}^{i_l} = \varphi'\left(u_{i_l}\right) w_{i_{l-1}}^{i_l}. \tag{9}$$

It is a random variable of order $1/\sqrt{n_{l-1}}$. Here and hereafter, we denote $\overset{l}{w}_{ij}$ by $w_{i_{l-1}}^{i_l}$, eliminating superfix l and using i_l and i_{l-1} instead of i and j. These index notations are convenient for showing that the corresponding w's belong to layer l.

We show how the square of the Euclidean length of $d\,\overset{l}{\boldsymbol{x}}$,

$$d\overset{l}{s}{}^2 = \sum_{l_i}\left(dx_{l_i}\right)^2, \tag{10}$$

is related to that of $d\,\overset{l-1}{\boldsymbol{x}}$. This relation can be seen from

$$d\,\overset{l}{s}{}^2 = \sum_{i_l, i_{l-1}, i'_{l-1}} B_{i_{l-1}}^{i_l} B_{i'_{l-1}}^{i_l}\, dx_{i_{l-1}} dx_{i'_{l-1}}. \tag{11}$$

For any pair i_{l-1} and i'_{l-1}, n_l random variables $B_{i_{l-1}}^{i_l} B_{i'_{l-1}}^{i_l}$ are iid for all i_l when $\overset{l-1}{\boldsymbol{x}}$ is fixed, so the law of large numbers guarantees that

$$\sum_{i_l} B_{i_{l-1}}^{i_l} B_{i'_{l-1}}^{i_l} = n_l \mathrm{E}\left[\varphi'\left(u_{i_l}\right)^2 w_{i_{l-1}}^{i_l} w_{i'_{l-1}}^{i_l}\right] + O_p\left(\frac{1}{\sqrt{n_l}}\right), \tag{12}$$

where E is the expectation with respect to the weights and biases and $O_p(1/\sqrt{n})$ represents small terms of

stochastic order $1/\sqrt{n}$. We use the mean field property that $\varphi'(u_{i_l})$ has the self-averaging property and the average of the product of $\varphi'(u_{i_l})^2$ and $w^{i_l}_{i_{l-1}} w^{i_l}_{i'_{l-1}}$ in equation (12) splits as

$$\mathrm{E}\left[\varphi'(u_{i_l})^2\right]\mathrm{E}\left[w^{i_l}_{i_{l-1}} w^{i_l}_{i'_{l-1}}\right]. \tag{13}$$

This is justified in appendix I. By putting

$$\overset{l}{\chi} = \sigma_l^2 \int \left\{\varphi'(\tau_l v)\right\}^2 Dv, \tag{14}$$

we have from equation (11)

$$d\,\overset{l}{s}{}^2 = \overset{l}{\chi}\, d\,\overset{l-1}{s^2}, \tag{15}$$

by using

$$\mathrm{E}\left[w^{i_l}_{i_{l-1}}\, w^{i_l}_{i'_{l-1}}\right] = \frac{\sigma_l^2}{n_l}\delta_{i_{l-1}i'_{l-1}}. \tag{16}$$

Here $\overset{l}{\chi}$ which depends on $\overset{l-1}{A}$, is the enlargement factor showing how $d\,\overset{l-1}{\boldsymbol{x}}$ is enlarged or reduced across layer l.

¿From the recursive relation (15), we have

$$d\,\overset{L}{s^2} = \chi_l^L\, d\,\overset{l-1}{s^2}, \tag{17}$$

$$\chi_l^L = \overset{L}{\chi}\,\overset{L-1}{\chi}\,\cdots\,\overset{l}{\chi}. \tag{18}$$

Assume that all the $\overset{l}{\chi}$ are equal. Then, it gives the Lyapunov exponent of dynamics equation (15). When it is larger than 1, the length diverges as the layers proceed, whereas when it is smaller than 1 the length decays to 0. The dynamics of $d\,\overset{l}{s^2}$ is chaotic when $\overset{l}{\chi} > 1$ (Poole et al, 2016). Interesting information processing takes place at the edge of chaos, where χ_l^L is nearly equal to 1 (Yang & Schoenholz, 2017).

3 FISHER INFORMATION OF DEEP NETWORKS AND NATURAL GRADIENT LEARNING

We study a regression model in which the output of layer L, $\overset{L}{\boldsymbol{x}} = \varphi(\overset{L}{\boldsymbol{u}})$,

$$\boldsymbol{y} = \overset{L}{\boldsymbol{x}} + \boldsymbol{\varepsilon}, \tag{19}$$

where $\boldsymbol{\varepsilon} \sim N(0, \boldsymbol{I})$ is a multivariate Gaussian random variable with mean 0 and identity covariance matrix $\boldsymbol{I}$. Then the probability of $\boldsymbol{y}$ given input $\boldsymbol{x}$ is

$$p(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{W}) = \frac{1}{\left(\sqrt{2\pi}\right)^{n_L}} \exp\left\{-\frac{1}{2}\left|\boldsymbol{y} - \overset{L}{\boldsymbol{x}}\right|^2\right\}, \tag{20}$$

where $\boldsymbol{W}$ consists of all the parameters $\overset{l}{\boldsymbol{w}}$, and $\overset{l}{b}$, $l = 1, \cdots, L$. The Fisher information matrix is given by

$$\boldsymbol{G} = \mathrm{E}_{\boldsymbol{x},\boldsymbol{y}}\left[\left(\partial_{\boldsymbol{W}} \log p\right)\left(\partial_{\boldsymbol{W}} \log p\right)\right], \tag{21}$$

where $\mathrm{E}_{\boldsymbol{x},\boldsymbol{y}}$ denotes the expectation with respect to randomly generated input $\boldsymbol{x}$ and resultant $\boldsymbol{y}$ and $\partial_{\boldsymbol{W}} = \partial/\partial \boldsymbol{W}$ is gradient with respect to $\boldsymbol{W}$. By using error vector $\boldsymbol{\varepsilon}$ in (19), we have

$$\partial_{\boldsymbol{W}} \log p = \boldsymbol{\varepsilon} \cdot \partial_{\boldsymbol{W}}\, \overset{L}{\boldsymbol{x}}. \tag{22}$$

For fixed $\boldsymbol{x}$, expectation with respect to $\boldsymbol{y}$ is replaced by that of $\boldsymbol{\varepsilon}$, where $\mathrm{E}[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}] = \boldsymbol{I}$. Hence, (21) is given by

$$\boldsymbol{G} = \mathrm{E}_{\boldsymbol{x}}\left[\sum_{i_L}\left\{\partial_{\boldsymbol{W}}\varphi(u_{i_L})\right\}\left\{\partial_{\boldsymbol{W}}\varphi(u_{i_L})\right\}\right]. \tag{23}$$

Here, we use the dyadic or tensor notation that $\boldsymbol{ab}$ implies a matrix $(a_i b_j)$, instead of vector-matrix notation $\boldsymbol{a}\boldsymbol{b}^T$ for column vectors.

Online learning is a method of modifying the current $\boldsymbol{W}$ such that the current loss

$$l = \frac{1}{2}\left|\boldsymbol{y} - \boldsymbol{x}_t^L\right|^2 \tag{24}$$

decreases, where $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ is the current input-output pair. The stochastic gradient decent method (proposed in Amari, 1967) uses the gradient of L to modify $\boldsymbol{W}$,

$$\Delta \boldsymbol{W} = -\eta \frac{\partial l}{\partial \boldsymbol{W}}. \tag{25}$$

Historically, the first simulation results applied to four-layer networks for pattern classification were given in a Japanese book (Amari, 1968). The minibatch method uses the average of $\partial l/\partial \boldsymbol{W}$ over minibatch samples.

The negative gradient is a direction to decrease the current loss but is not steepest in a Riemannian manifold. The true steepest direction is given by

$$\tilde{\nabla} l = \boldsymbol{G}^{-1} \frac{\partial l}{\partial \boldsymbol{W}}, \tag{26}$$

which is called the natural or Riemannian gradient (Amari, 1998). The natural gradient method is given by

$$\Delta \boldsymbol{W} = -\eta \tilde{\nabla} l. \tag{27}$$

It is known to be Fisher efficient for estimating $\boldsymbol{W}$ (Amari, 1998). Although it gives excellent performances, the inversion of $\boldsymbol{G}$ is computationally very difficult.

To avoid difficulty, the quasi-diagonal natural gradient method is proposed in Ollivier (2015) and is shown to be very efficient in Marcereau-Caron and Ollivier

(2016). A recent proposal (Ollivier, 2017) looks very promising for realizing natural gradient learning. The present paper analyzes the structure of the Fisher information matrix. It will give a justification of the quasi-diagonal natural gradient method. By using it, we propose a new method of realizing natural gradient learning without the burden of inverting $\boldsymbol{G}$.

4 STRUCTURE OF FISHER INFORMATION MATRIX

To calculate elements of $\boldsymbol{G}$, we use a new notation combining connection weights $\boldsymbol{w}$ and bias b into one vector,

$$\overset{l}{\boldsymbol{w}}{}^{*}=\left(\overset{l}{\boldsymbol{w}},\overset{l}{b}\right). \tag{28}$$

For the i_l-th unit of layer l, it is

$$\boldsymbol{w}_{i_l}^{*}=\left(w_{i_{l-1}}^{i_l},b^{i_l}\right). \tag{29}$$

For $l>m$, we have the recursive relation

$$\frac{\partial \overset{l}{\boldsymbol{x}}}{\partial \overset{m}{\boldsymbol{w}}{}^{*}}=\frac{\partial \overset{l}{\boldsymbol{x}}}{\partial \overset{l-1}{\boldsymbol{x}}}\frac{\partial \overset{l-1}{\boldsymbol{x}}}{\partial \overset{m}{\boldsymbol{w}}{}^{*}}=\overset{l}{\boldsymbol{B}}\frac{\partial \overset{l-1}{\boldsymbol{x}}}{\partial \overset{m}{\boldsymbol{w}}{}^{*}}. \tag{30}$$

Starting from $l=L$ and using

$$\frac{\partial \overset{m}{\boldsymbol{x}}}{\partial \overset{m}{\boldsymbol{w}}{}^{*}}=\varphi'\left(\overset{m}{\boldsymbol{u}}\right)\overset{m-1}{\boldsymbol{x}}, \tag{31}$$

we have

$$\frac{\partial \overset{L}{\boldsymbol{x}}}{\partial \overset{m}{\boldsymbol{w}}{}^{*}}=\overset{L}{\boldsymbol{B}}\cdots\overset{m+1}{\boldsymbol{B}}\varphi'\left(\overset{m}{\boldsymbol{u}}\right)\overset{m-1}{\boldsymbol{x}}. \tag{32}$$

Put

$$\boldsymbol{B}_{m+1}^{L}=\overset{L}{\boldsymbol{B}}\cdots\overset{m+1}{\boldsymbol{B}}, \tag{33}$$

which is a product of $L-(m-1)$ matrices. The elements of $\boldsymbol{B}_{m+1}^{L}$ are denoted by $B_{i_m}^{i_L}$.

We calculate the Fisher information $\boldsymbol{G}$ given in equation (23). The elements of $\boldsymbol{G}$ with respect to layers l and m are written as

$$\boldsymbol{G}\left(\overset{m}{\boldsymbol{w}}{}^{*},\overset{l}{\boldsymbol{w}}{}^{*}\right)=\mathrm{E}_{\boldsymbol{x}}\left[\frac{\partial \overset{L}{\boldsymbol{x}}}{\partial \overset{m}{\boldsymbol{w}}{}^{*}}\cdot\frac{\partial \overset{L}{\boldsymbol{x}}}{\partial \overset{l}{\boldsymbol{w}}{}^{*}}\right], \tag{34}$$

where $\cdot$ denotes the innor product with respect to $\overset{L}{\boldsymbol{x}}$. The (i_m,i_{m-1}) elements of $\partial \overset{L}{\boldsymbol{x}}/\partial \overset{m}{\boldsymbol{w}}{}^{*}$ are, for fixed i_L,

$$B_{i_m}^{i_L}\varphi'\left(u_{i_m}\right)x_{i_{m-1}}. \tag{35}$$

Hence, (34) is written in the component form as

$$\left[\boldsymbol{G}\left(\overset{m}{\boldsymbol{w}}{}^{*},\overset{l}{\boldsymbol{w}}{}^{*}\right)\right]_{i_{m-1}i_{l-1}}^{i_m i_l}=\sum_{i_L}B_{i_m}^{i_L}B_{i_l}^{i_L}\varphi'\left(u_{i_m}\right)\varphi'\left(u_{i_l}\right)x_{i_{m-1}}x_{i_{l-1}}. \tag{36}$$

We first consider the case $m=l$, that is, two neurons are in the same layer m. The following lemma is useful for evaluating $\sum B_{i_m}^{i_L}B_{i'_m}^{i_L}$.

Domino Lemma We assume that all n_l are of order n.

$$\sum_{i_L,i'_L}\delta_{i_L i'_L}B_{i_m}^{i_L}B_{i'_m}^{i'_L}=\chi_{m+1}^{L}\delta_{i_m i'_m}+O_p\left(\frac{1}{\sqrt{n}}\right). \tag{37}$$

Proof. We first prove the case with $m=L-1$. We have

$$\sum\delta_{i_L i'_L}B_{i_{L-1}}^{i_L}B_{i'_{L-1}}^{i'_L}=\sum_{i_L}\left\{\varphi'\left(u_{i_L}\right)\right\}^2 w_{i_{L-1}}^{i_L}w_{i'_{L-1}}^{i_L}. \tag{38}$$

When $i_{L-1}=i'_{L-1}$, this is a sum of n_L iid random variables $\left\{\varphi'\left(u_{i_L}\right)\right\}^2\left(w_{i_{L-1}}^{i_L}\right)^2$, when input $\boldsymbol{x}$ is fixed. Therefore, the law of large numbers guarantees that, as n_L goes to infinity, their sum converges to the expectation,

$$n_L\mathrm{E}_{\boldsymbol{x}}\left[\left\{\varphi'\left(u_{i_L}\right)\right\}^2\left(w_{i_{L-1}}^{i_L}\right)^2\right]=\overset{L}{\chi} \tag{39}$$

under the mean field approximation for any i_L. For fixed $i_{L-1}\neq i'_{L-1}$, the right-hand side of equation (38) is also a sum of iid variables with mean 0. Hence, its mean is 0. We evaluate its variance, proving that the variance is

$$n_L\mathrm{E}\left[\left\{\varphi'\left(u_{i_L}\right)\right\}^4\left(w_{i_{L-1}}^{i_L}\right)^2\left(w_{i'_{L-1}}^{i_L}\right)^2\right] \tag{40}$$

which is of order $1/n_L$, because $\mathrm{E}\left[\left(w_{i_{L-1}}^{i_L}\right)^2\right]$ is of order $1/n_L$. Hence we have

$$\sum\delta_{i_L i'_L}B_{i_{L-1}}^{i_L}B_{i'_{L-1}}^{i'_L}=\overset{L}{\chi}\,\delta_{i_{L-1}i'_{L-1}}+O_p\left(\frac{1}{\sqrt{n_L}}\right). \tag{41}$$

When $m<L-1$, we repeat the process $L-1,\cdots$. Then $\delta_{i_L i'_L}$ in the left-hand side of equation (37) propagates to give $\delta_{i_m i'_m}$ like the domino effect, leaving multiplicative factors $\overset{l}{\chi}$. This proves the theorem. □

Remark: The domino lemma holds irrespective of $n_l>n_{l-1}$ or $n_l<n_{l-1}$, provided they are large. However, matrix $\boldsymbol{B}_m^L$ is not of full rank, its rank being $\min\{n_L,\cdots,n_m\}$.

By using this result, we evaluate off-diagonal blocks of $\boldsymbol{G}$ under the mean field approximation (39).

Theorem 1. The Fisher information matrix $\boldsymbol{G}$ is unit-wise diagonal except for terms of stochastic order $O_p(1/\sqrt{n})$.

Proof. We first calculate the off-diagonal blocks of the Fisher information matrix within the same layers. The Fisher information submatrix within layer m is

$$\boldsymbol{G}\left(\overset{m}{\boldsymbol{w}}{}^{*}, \overset{m}{\boldsymbol{w}}{}^{'*}\right) = E_{\boldsymbol{x}}\Big[\sum \delta_{i_L i'_L} B_{i_m}^{i_L} B_{i'_m}^{i_L} \times \varphi'\left(u_{i_m}\right)\varphi'\left(u_{i'_m}\right) x_{i_{m-1}} x_{i'_{m-1}}\Big], \tag{42}$$

which are elements of submatrix of $\boldsymbol{G}$ corresponding to neurons i_m and i'_m both in the same layer m. By the domino lemma, we have

$$G\left(\overset{m}{\boldsymbol{w}}{}^{*}, \overset{m}{\boldsymbol{w}}{}^{'*}\right) = \mathrm{E}_{\boldsymbol{x}}\left[\chi_m^L \left\{\varphi'\left(u_{i_m}\right)\right\}^2 x_{i_{m-1}} x_{i'_{m-1}}\right] \delta_{i_m i'_m}, \tag{43}$$

except for terms of order $1/\sqrt{n}$. This shows that the submatrix is unit-wise block diagonal: That is, the blocks of different neurons i_m and i'_m $(i_m \neq i'_m)$ are 0 except for terms of order $1/\sqrt{n}$.

We next study the blocks of different layers l and m $(m < l)$. We have

$$B_{i_m}^{i_L} = \sum_{i_l} B_{i_l}^{i_L} B_{i_m}^{i_l}. \tag{44}$$

By using the domino lemma, $\boldsymbol{G}$ is written as

$$\mathrm{E}_{\boldsymbol{x}}\left[\chi_l^L B_{i_m}^{i_l} \varphi'\left(u_{i_l}\right)\varphi'\left(u_{i_m}\right) \overset{l-1}{\boldsymbol{x}}\, \overset{m-1}{\boldsymbol{x}}\right]. \tag{45}$$

When $m = l-1$,

$$B_{i_{m-1}}^{i_l} = \varphi'\left(u_{i_l}\right) w_{i_{m-1}}^{i_l} \tag{46}$$

and hence it is of order $1/\sqrt{n}$. In general, $B_{i_m}^{i_l}$ is a sum of n^{l-m} 0 mean iid random variables with variance of order $1/n^{l-m+1}$. Hence, its mean is 0 and variance is of order $1/n$, proving that (45) is of order $1/\sqrt{n}$. □

Inspired from this, we define a new metric $\boldsymbol{G}^*$ as an approximation of $\boldsymbol{G}$, such that all the off-diagonal block terms of $\boldsymbol{G}$ are discarded, putting them equal to 0. We study the natural (Riemannian) gradient method which uses $\boldsymbol{G}^*$ as the Riemannian metric. Note that $\boldsymbol{G}^*$ is an approximation of $\boldsymbol{G}$, $\boldsymbol{G}$ tending to $\boldsymbol{G}^*$ for $n \to \infty$ in the max-norm, but $\boldsymbol{G}^{*-1}$ is not a good approximation to $\boldsymbol{G}^{-1}$. This is because the max-norm of a matrix is not sub-multiplicative. In other words, when $\boldsymbol{G}$ is approximately block-diagonal, $\boldsymbol{G}^2$ and $\boldsymbol{G}^{-1}$ do not have this property.

The present study focuses on the approximated metric $\boldsymbol{G}^*$. It enables us to give an explicit form of the Fisher information matrix, directly applicable to natural gradient methods, as follows.

5 UNIT-WISE FISHER INFORMATION

Because $\boldsymbol{G}^*$ is unit-wise block-diagonal, it is enough to calculate the Fisher information matrices of single units. We assume that its input vector $\boldsymbol{x}$ is subject to $N(0, \boldsymbol{I})$. This does not hold in general. However, it holds approximately for a randomly connected resnet, as is shown in the next section.

Let us introduce a new $(n+1)$-dimensional vectors for a single unit:

$$\boldsymbol{w}^* = (\boldsymbol{w}, w_0), \tag{47}$$
$$\boldsymbol{x}^* = (\boldsymbol{x}, x_0), \tag{48}$$

where $w_0 = b$ and $x_0 = 1$. Then, the output of the unit is $\varphi(u) = \varphi(\boldsymbol{w}^* \cdot \boldsymbol{x}^*)$, $u = \boldsymbol{w}^* \cdot \boldsymbol{x}^*$. The Fisher information is an $(n+1) \times (n+1)$ matrix written as

$$\boldsymbol{G} = \mathrm{E}_{\boldsymbol{x}}\left[\left\{\varphi'(u)\right\}^2 \boldsymbol{x}^* \boldsymbol{x}^*\right]. \tag{49}$$

We introduce a set of new $n+1$ orthonormal basis vectors in the space of $\boldsymbol{x}^* = (\boldsymbol{x}, x_0)$ as

$$\boldsymbol{e}_0^* = (0, \cdots, 0, 1), \tag{50}$$
$$\boldsymbol{e}_i^* = (\boldsymbol{a}_i, 0), \quad i = 1, 2, \cdots, n-1, \tag{51}$$
$$\boldsymbol{e}_n^* = \frac{1}{w}(\boldsymbol{w}, 0), \quad w^2 = \boldsymbol{w} \cdot \boldsymbol{w}, \tag{52}$$

where $\boldsymbol{a}_i, i = 1, \cdots n$, are arbitrary orthogonal unit vectors, satisfying $\boldsymbol{a}_i \cdot \boldsymbol{w} = 0$, $\boldsymbol{a}_i \cdot \boldsymbol{a}_j = \delta_{ij}$. That is, $\{\boldsymbol{e}_1^*, \cdots, \boldsymbol{e}_n^*\}$ is a rotation of $\{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_n\}$ and we put $\boldsymbol{e}_0 = \boldsymbol{e}_0^*$.

Here $\{\boldsymbol{e}_i^*\}, i = 0, 1, \cdots, n, n+1$ are mutually orthogonal unit vectors and $\boldsymbol{e}_n^*$ is the unit vector in the direction of $\boldsymbol{w}$. Since $\boldsymbol{x}^*$ and $\boldsymbol{w}^*$ are represented in the new basis as

$$\boldsymbol{x}^* = \sum_{i=0}^{n} x_i^* \boldsymbol{e}_i^*, \quad \boldsymbol{w}^* = b\boldsymbol{e}_0^* + w\boldsymbol{e}_n^*, \tag{53}$$

we have

$$\boldsymbol{G} = \mathrm{E}\left[\left\{\varphi'\left(\boldsymbol{w}^* \cdot \boldsymbol{x}^*\right)\right\}^2 \boldsymbol{x}^* \boldsymbol{x}^*\right]. \tag{54}$$

Moreover, $(x_1^*, \cdots, x_n^*)$ are orthogonal transformation of $\boldsymbol{x} = (x_1, \cdots, x_n)$. Hence, $x_i^*, i = 1, \cdots, n$, are

jointly independent Gaussian, subject to $N(0,1)$, and $x_0^* = 1$.

In order to obtain $\boldsymbol{G}$, let us put

$$\boldsymbol{G} = \sum_{i,j=0}^{n} A_{ij} \boldsymbol{e}_i^* \boldsymbol{e}_j^* \tag{55}$$

in the dyadic notation. Then, the coefficients A_{ij} are given by

$$A_{ij} = \boldsymbol{e}_i^* \boldsymbol{G} \boldsymbol{e}_j^*, \tag{56}$$

which are (i,j) elements of $\boldsymbol{G}$ in the coordinate system $\{\boldsymbol{e}_i^*\}$. From $\boldsymbol{w}^* \cdot \boldsymbol{x}^* = w x_n + w_0$ and equation (54), we have

$$\begin{aligned} A_{00} &= \int \left\{\varphi'\left(w x_n^* + w_0\right)\right\}^2 D x_n^*, & (57)\\ A_{0n} &= \int x_n^* \left\{\varphi'\left(w x_n^* + w_0\right)\right\}^2 D x_n^*, & (58)\\ A_{nn} &= \int x_n^{*2} \left\{\varphi'\left(w x_n^* + w_0\right)\right\}^2 D x_n^*, & (59) \end{aligned}$$

which depend on $(\boldsymbol{w}, w_0)$. We further have, for $i = 1, \cdots, n-1$,

$$\begin{aligned} A_{ii} &= \boldsymbol{e}_i^* \boldsymbol{G} \boldsymbol{e}_i^* = A_{00}, & (60)\\ A_{ij} &= \boldsymbol{e}_i^* \boldsymbol{G} \boldsymbol{e}_j^* = 0 \quad (j \neq i), & (61)\\ A_{i0} &= \boldsymbol{e}_i^* \boldsymbol{G} \boldsymbol{e}_0^* = 0 \quad (i \neq n). & (62) \end{aligned}$$

¿From these we obtain $\boldsymbol{G}$ in the dyadic form

$$\begin{aligned} \boldsymbol{G} &= A_{00} \sum_{i=0}^{n} \boldsymbol{e}_i^* \boldsymbol{e}_i^* + \left(A_{nn} - A_{00}\right) \boldsymbol{e}_n^* \boldsymbol{e}_n^* \\ &\quad + A_{0n} \left(\boldsymbol{e}_0^* \boldsymbol{e}_n^* + \boldsymbol{e}_n^* \boldsymbol{e}_0^*\right). \end{aligned} \tag{63}$$

The elements of $\boldsymbol{G}$ in the basis $\{\boldsymbol{e}_1^*, \cdots, \boldsymbol{e}_n^*, \boldsymbol{e}_0^*\}$ are

$$\boldsymbol{G} = \left[\begin{array}{ccc:c} A_{00} & & 0 & \\ & \ddots & & 0 \\ 0 & & A_{00} & \\ \hdashline & 0 & & \begin{matrix} A_{nn} & A_{n0} \\ A_{n0} & A_{00}. \end{matrix} \end{array}\right], \tag{64}$$

which shows that $\boldsymbol{G}$ is a sum of a diagonal matrix and a rank 2 matrix.

The inverse of $\boldsymbol{G}$ has the same block form as equations (63) and (64). Note that

$$\begin{aligned} \sum \boldsymbol{e}_i^* \boldsymbol{e}_i^* &= \boldsymbol{I}, & (65)\\ \boldsymbol{e}_n^* \boldsymbol{e}_n^* &= \frac{1}{w^2} \boldsymbol{w}\boldsymbol{w}, & (66)\\ \boldsymbol{e}_0^* \boldsymbol{e}_n^* + \boldsymbol{e}_n^* \boldsymbol{e}_0^* &= \frac{1}{w} \left(\boldsymbol{e}_0 \tilde{\boldsymbol{w}} + \tilde{\boldsymbol{w}} \boldsymbol{e}_0\right), & (67) \end{aligned}$$

where $\tilde{\boldsymbol{w}} = (\boldsymbol{w}, 0)$.

By using these relations, $\boldsymbol{G}$ is expressed in the original basis as

$$\boldsymbol{G} = \sum A_{00}\, \boldsymbol{I} + \frac{\left(A_{nn} - A_{00}\right)}{w^2} \tilde{\boldsymbol{w}}\tilde{\boldsymbol{w}} + \frac{A_{0n}}{w} \left(\boldsymbol{e}_0 \tilde{\boldsymbol{w}} + \tilde{\boldsymbol{w}} \boldsymbol{e}_0\right). \tag{68}$$

The inverse of $\boldsymbol{G}$ has also the same form, so we have an explicit form of $\boldsymbol{G}^{-1}$

$$\begin{aligned} \boldsymbol{G}^{-1} &= \bar{A}_{00} \boldsymbol{I} + \frac{X}{w^2} \tilde{\boldsymbol{w}}\tilde{\boldsymbol{w}} + \frac{Y}{w} \left(\boldsymbol{e}_0^* \tilde{\boldsymbol{w}} + \tilde{\boldsymbol{w}} \boldsymbol{e}_0^*\right) & (69)\\ &\quad + Z \boldsymbol{e}_0^* \boldsymbol{e}_0^*, & (70) \end{aligned}$$

where

$$\begin{aligned} \bar{A}_{00} &= \frac{1}{A_{00}}, & X &= \frac{1}{D} A_{00} - \bar{A}_{00}, & (71)\\ Y &= \frac{-A_{n0}}{D}, & Z &= \frac{A_{nn}}{D} - \bar{A}_{00}, & (72)\\ D &= A_{00} A_{nn} - A_{n0}^2. & & & (73) \end{aligned}$$

By using the above equations, $\boldsymbol{G}^{-1}\boldsymbol{x}^*$ is obtained explicitly, so we do not need to calculate back-propagated $\boldsymbol{G}$ and its inverse for the natural gradient update of $\boldsymbol{W}$.

The natural gradient method for each unit is written by using the back-propagated error e as

$$\Delta \boldsymbol{w}^* = -\eta e \boldsymbol{G}^{-1} \boldsymbol{x}^*, \tag{74}$$

which splits as

$$\begin{aligned} \Delta \boldsymbol{w} &= -\eta e \left[\bar{A}_{00} \boldsymbol{x} + \left(\frac{X}{\boldsymbol{w}^2} \boldsymbol{w} \cdot \boldsymbol{x} + \frac{Y}{w}\right) \boldsymbol{w}\right], & (75)\\ \Delta w_0 &= -\eta e \left(\bar{A}_{00} + \frac{\boldsymbol{w} \cdot \boldsymbol{x}}{w} Y + Z\right) w_0. & (76) \end{aligned}$$

We can implement the unit-wise natural gradient method using equations (75) and (76) without calculating $\boldsymbol{G}^{*-1}$. We will see that a residual network automatically makes the input $\boldsymbol{x}$ to each layer be subject to a 0-mean Gaussian distribution approximately.

Obviously, $\boldsymbol{W}$ is no more random Gaussian with mean 0 after learning. However, since the unit-wise natural gradient proposed here is computationally so easy, it is worth trying for practical applications even in the process of learning.

Except for a rank 1 term $\tilde{\boldsymbol{w}}\tilde{\boldsymbol{w}} = (w_i w_j)$ and bias terms $\boldsymbol{e}_0 \tilde{\boldsymbol{w}} + \tilde{\boldsymbol{w}} \boldsymbol{e}_0$, $\boldsymbol{G}^{-1}$ is a diagonal matrix. In other words, as is seen in equation (67), it is diagonal except for a raw and column corresponding to the bias terms and the rank 1 term $\tilde{\boldsymbol{w}}\tilde{\boldsymbol{w}}$. Except for the rank 1 term $\tilde{\boldsymbol{w}}\tilde{\boldsymbol{w}}$, it has the same structure as that of the quasi-diagonal matrix of Ollivier (2015).

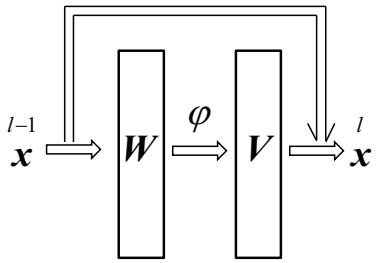

Figure 2: Residual network

6 FISHER INFORMATION OF RESIDUAL NETWORK

The residual network has direct paths from its input to output in each layer. We treat the following block of layer l: The layer l transforms input $\overset{l-1}{\boldsymbol{x}}$ to output $\overset{l}{\boldsymbol{x}}$ by

$$\overset{l}{x}_i = \sum_j \overset{l}{v}_{ij}\, \varphi\left(\overset{l}{u}_j\right) + \alpha\, \overset{l-1}{x}_i, \tag{77}$$

$$\overset{l}{u}_j = \sum_k \overset{l}{w}_{jk}\, \overset{l-1}{x}_k + \overset{l}{b}_j \tag{78}$$

(see Figure 2). Here $\alpha \leq 1$ is a decay factor, ($\alpha = 1$ is conventionally used), and $\overset{l}{v}_{ij}$ are randomly generated iid Gaussian variables subject to $N(0, \sigma_v^2/n)$.

We show how the activity develops in a residual network (Yang and Schoenholtz, 2017). We easily have the recursive relation,

$$\overset{l}{A} = \sigma_v^2 \bar{A}^l + \alpha^2\, \overset{l-1}{A}, \tag{79}$$

where

$$\bar{A}^l = \int \left\{\varphi\left(\tau_l v\right)\right\}^2 Dv \tag{80}$$

Eq (79) shows that $\overset{l}{A}$ diverges to infinity as l increase when $\alpha \geq 1$. Therefore, we recommend to use $\alpha < 1$.

The layer l consists of two sublayers. One is the ordinary neural network with weights $\overset{l}{w}_{jk}$, bias $\overset{l}{b}_j$ and activation function φ. The other is a linear network that randomizes the outputs $\varphi\left(u_j\right)$ of the first layer, transforming them to asymptotically independent 0-mean Gaussian random variables. Therefore, mean 0 quasi independent Gaussianity is guaranteed for $\overset{l}{\boldsymbol{x}}$. Since the second linear network is used for the purpose making output $\overset{l}{\boldsymbol{x}}$ subject to 0-mean independent Gaussian distributions, we fix them throughout the learning. That is, $\left\{\overset{l}{w}_{ij}, \overset{l}{b}_i\right\}$ are subject only to stochastic gradient learning. Therefore, we study the Fisher information with respect to $\left\{\overset{l}{w}_{ij}, \overset{l}{b}_i\right\}$ only. It is redundant to train both v_{ij} and w_{ij}. The role of v_{ij} is to Gaussianize the outputs of layers. We recommend to fix $\overset{l}{v}_{ij}$ throughout learning processes once they are randomly assigned in the initial stage.

We calculate the following recursive formula,

$$\frac{\partial\, \overset{l}{x}_i}{\partial\, \overset{m}{w}_{st}} = \sum_{j,k} \overset{l}{v}_{ij}\, \varphi'\left(\overset{l}{u}_j\right) w_{jk}^l \frac{\partial\, \overset{l-1}{x}_k}{\partial\, \overset{m}{w}_{st}} + \alpha \frac{\partial\, \overset{l-1}{x}_i}{\partial\, \overset{m}{w}_{st}} \tag{81}$$

$$= \sum_k \overset{l}{B}_{ik}\, \frac{\partial\, \overset{l-1}{x}_k}{\partial\, \overset{m}{w}_{st}}, \tag{82}$$

where

$$\overset{l}{B}_{ik} = \sum_j \overset{l}{v}_{ij}\, \varphi'\left(\overset{l}{u}_j\right) \overset{l}{w}_{jk} + \alpha \delta_{ik} \tag{83}$$

in the case of a residual net. Note that $\overset{l}{B}_{ik}$ is of order $1/\sqrt{n}$ when $i \neq k$, and

$$\overset{l}{B}_{ii} = \alpha + O_p(1/\sqrt{n}). \tag{84}$$

¿From this we have

$$\frac{\partial\, \overset{L}{x}_i}{\partial\, \overset{m}{w}_{st}} = \sum_k B_k^i\, \overset{m}{v}_{ks}\, \varphi'\left(\overset{m}{u}_s\right) \overset{m-1}{x}_t, \tag{85}$$

$$G\left(\overset{m}{w}_{st}, \overset{l}{w}_{s't'}\right) = \mathrm{E}_{\boldsymbol{x}}\Big[\sum_i B_{k_m}^i B_{k_l}^i\, \overset{m}{v}_{k_m s} \overset{l}{v}_{k_l s'} \tag{86}$$

$$\times\, \varphi'\left(\overset{m}{u}_s\right) \varphi'\left(\overset{l}{u}_{s'}\right) \overset{m-1}{x}_t\, \overset{l-1}{x}_{t'}\Big]. \tag{87}$$

Here we again use the domino theorem, where previous χ is replaced by

$$\bar{\chi} = \sigma_v^2 \chi + \alpha. \tag{88}$$

Since $\overset{m}{v}_{ks}$ and $\overset{l}{v}_{k's'}$ are independent when $m \neq l$, $G\left(\overset{m}{w}_{st}, \overset{l}{w}_{s't'}\right)$ is of order $1/\sqrt{n}$. This is true when $m = l$, $s \neq s'$. So we have the following theorem.

Theorem 2. The Fisher information matrix $\boldsymbol{G}$ of a residual net is unit-wise diagonal to within terms of order $1/\sqrt{n}$.

We suggest the following approximate learning algorithm for a residual network with the ReLU activation function:

1. Fix σ_v^2 and σ_w^2 and $\alpha < 1$.

2. Given a training example $(\boldsymbol{y}_t, \boldsymbol{x}_t)$, calculate the back-propagated error e_{i_m} for each unit i_m of the m-th layer.

3. Using the current $\boldsymbol{w}_{i_m}^*$, calculate $\bar{A}_\infty, X, Y$ and Z from equations (71)–(73) and Appendix II.

4. Update the current $\boldsymbol{w}_{i_m}^*$ by using equations (75)–(76).

5. We may use the Polyak averaging (Polyak & Juditsky, 1992) after learning.

It is interest to compare the result of the present algorithm with that of the quasi-diagonal method (Marceau-Caron & Ollivier, 2016).

7 CONCLUSIONS

The paper applies the statistical neurodynamical method for studying the Fisher information matrix of deep random networks. The Fisher information is calculated from back-propagated errors. The main result of the paper is to show that the Fisher information matrix in a large random network is neuron-wise block diagonalized approximately. This justifies the unit-wise natural gradient method (Ollivier, 2015). The unit-wise Fisher information is a tensor product of the Fisher information matrices of single neurons. We calculated the Fisher information and its inverse explicitly, showing its peculiar structure. The present paper is purely theoretical, waiting for its applications to practical networks.

APPENDIX I: SELF-AVERAGING PROPERTY

Let us treat a simple case

$$u = \sum w_i x_i, \tag{89}$$

where x_i are fixed and $w_i \sim N\left(0, \sigma^2/n\right)$. We consider

$$E\left[f(u) w_i w_j\right], \tag{90}$$

where we put $f(u) = \{\varphi'(u)\}^2$ and $i_{l-1} = i, i'_{l-1} = j$. Put

$$\tilde{u} = \sum_{i \neq 1,2} w_i x_i. \tag{91}$$

Then

$$\begin{aligned} u &= \tilde{u} + w_1 x_1 + w_2 x_2, & (92)\\ f(u) &= f\left(\tilde{u}\right) + f'\left(\tilde{u}\right)\left(w_i x_i + w_j x_j\right). & (93) \end{aligned}$$

We have, neglecting higher-order terms,

$$\begin{aligned} E\left[f(u) w_i w_j\right] = E\left[f\left(\tilde{u}\right) w_i w_j\right] + E\big[f'\left(\tilde{u}\right) w_i w_j & \quad (94)\\ \times \left(w_i x_i + w_j x_j\right)\big]. & \quad (95) \end{aligned}$$

Since $\tilde{u}$ and $w_1 w_2$ are independent,

$$\begin{aligned} E\left[f(u) w_i w_j\right] &= E\left[f\left(\tilde{u}\right)\right] E\left[w_i w_j\right] & (96)\\ &= E\left[f(u)\right] E\left[w_i w_j\right] & (97) \end{aligned}$$

except for higher-order terms.

APPENDIX II: FISHER INFORMATION FOR RELU

The ReLU activation function is given by

$$\varphi(u) = \begin{cases} u, & u > 0, \\ 0, & u \leq 0. \end{cases} \tag{98}$$

We calculate $A_{ij}\left(\boldsymbol{w}, w_0\right)$ given by equations (57)–(59). Since

$$\varphi'\left(w x_1^* + w_0\right) = \begin{cases} 1, & w x_1^* + w_0 > 0, \\ 0, & \text{otherwise}, \end{cases} \tag{99}$$

we have

$$\begin{aligned} A_{00} &= \frac{1}{\sqrt{2\pi}} \int_{-\frac{w_0}{w}}^{\infty} \exp\left\{-\frac{u^2}{2}\right\} du & (100)\\ &= \operatorname{erf}\left(\frac{w_0}{w}\right), & (101)\\ \operatorname{erf}(u) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{u} \exp\left\{-\frac{u^2}{2}\right\} dv. & (102) \end{aligned}$$

Similarly,

$$\begin{aligned} A_{0n} &= \frac{1}{\sqrt{2\pi}} \int_{-\frac{w_0}{w}}^{\infty} u \exp\left\{-\frac{u^2}{2}\right\} du & (103)\\ &= \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{1}{2}\left(\frac{w_0}{w}\right)^2\right\} & (104)\\ A_{nn} &= \frac{1}{\sqrt{2\pi}} \int_{-\frac{w_0}{w}}^{\infty} u^2 \exp\left\{-\frac{u^2}{2}\right\} du & (105)\\ &= \operatorname{erf}\left(\frac{w_0}{w}\right) - \frac{1}{\sqrt{2\pi}} \frac{w_0}{w} \exp\left\{-\frac{1}{2}\left(\frac{w_0}{w}\right)^2\right\}. & (106) \end{aligned}$$

References

S. Amari (1967). Theory of Adaptive Pattern Classifiers. *IEEE Trans.* **EC-16**(3):299–307.

S. Amari (1968). *Information Theory II, Geometrical Theory of Information (in Japanese).* Kyoritsu Shuppan.

S. Amari (1998). Natural Gradient Works Efficiently in Learning. *Neural Computation.* **10**(2):251–276.

S. Amari, R. Karakida, and M. Oizumi (2018). Statistical neurodynamics of deep networks I, Geometry of signal spaces. arXiv.

R. Grosse, and J. Martens (2016). *A Kronecker-factored approximate Fisher matrix for convolution layers.* ICML.

R. Karakida, S. Akaho, and S. Amari (2018). Universal statistics of Fisher information in deep neural networks: Mean field approach. arXiv.org/abs/1806.01316.

T. Kurita (1994). Iterative weighted least squares algorithms for neural networks classifiers. *New Generation Computing.* **12**:3750394.

G. Marceau-Caron, and Y. Ollivier (2016). Practical Riemannian neural networks. arXiv:1602.08007.

J. Martens (2017). New insight and perspective on the natural gradient method. arXiv:1412.1193v9.

Y. Ollivier (2015). Riemannian metrics for neural networks, I: Feedforward networks *Information and Inference.* **4**:108–153.

Y. Ollivier (2017). True asymptotic natural gradient optimization. arXiv:1712.08449v1.

R. Pascanu, and Y. Bengio (2013). Revisiting natural gradient for deep networks. arXiv:1301.3584v7.

B. T. Polyak, and A. B. Juditsky (1992). Acceleration of stochastic approximation by averaging. *SIAM J. on Control and Optimization.* **30**:838–855.

B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein and S. Ganguli (2016). Exponential expressivity in deep neural networks. *In Proc. NIPS*, 3360–3368.

S. S. Schoenholz, J. Gilmer, S. Ganguli, and J. Sohl-Dickstein (2016). Deep information propagation. *ICLR'2017.* arXiv:1611.01232.

G. Yang, S. Schoenholz (2017). Mean field residual networks: On the edge of chaos. *Proc. NIPS*, 2865–2873.