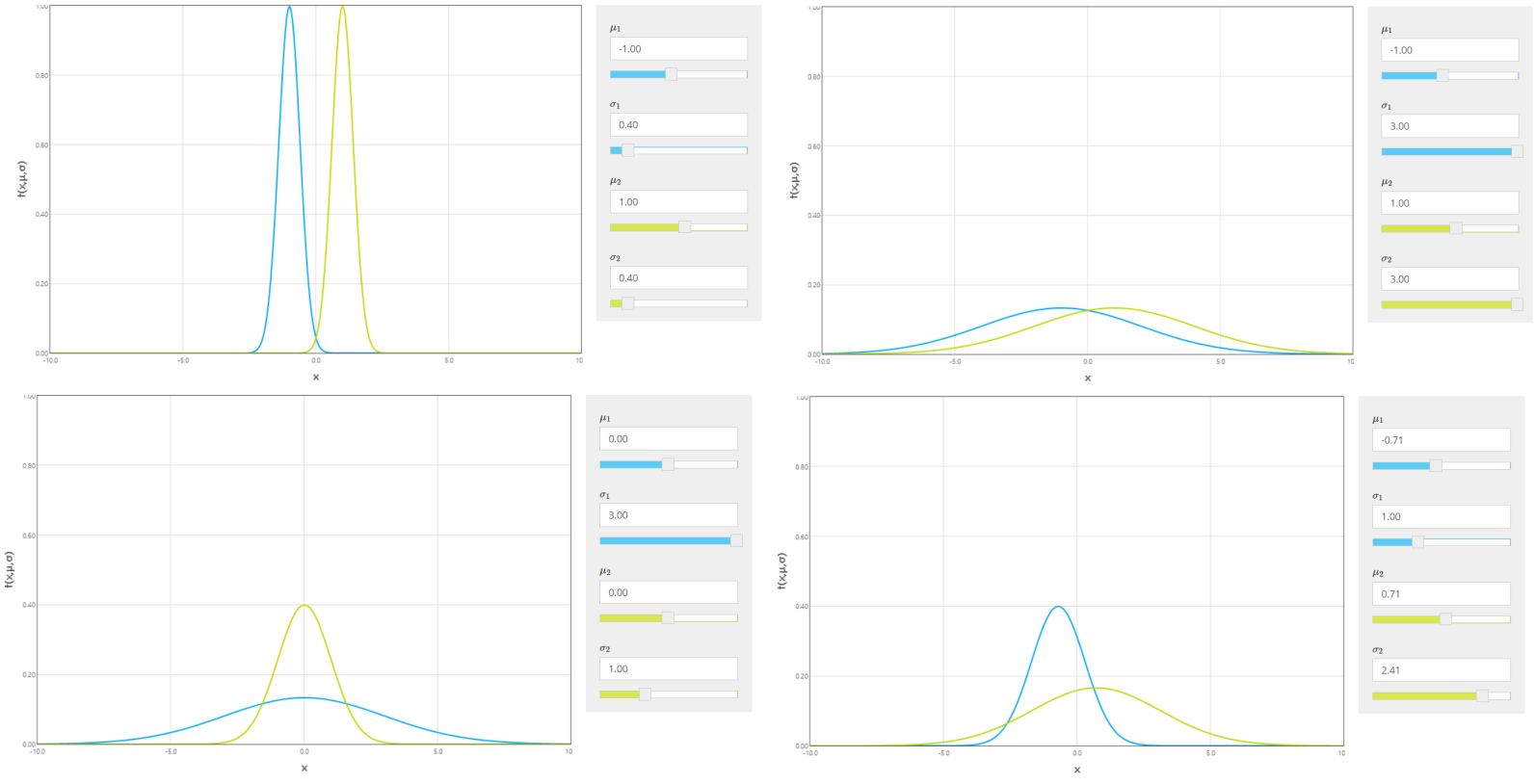
New Insights and Perspectives on the Natural Gradient Method

Yoonho Lee

Department of Computer Science and Engineering
Pohang University of Science and Technology

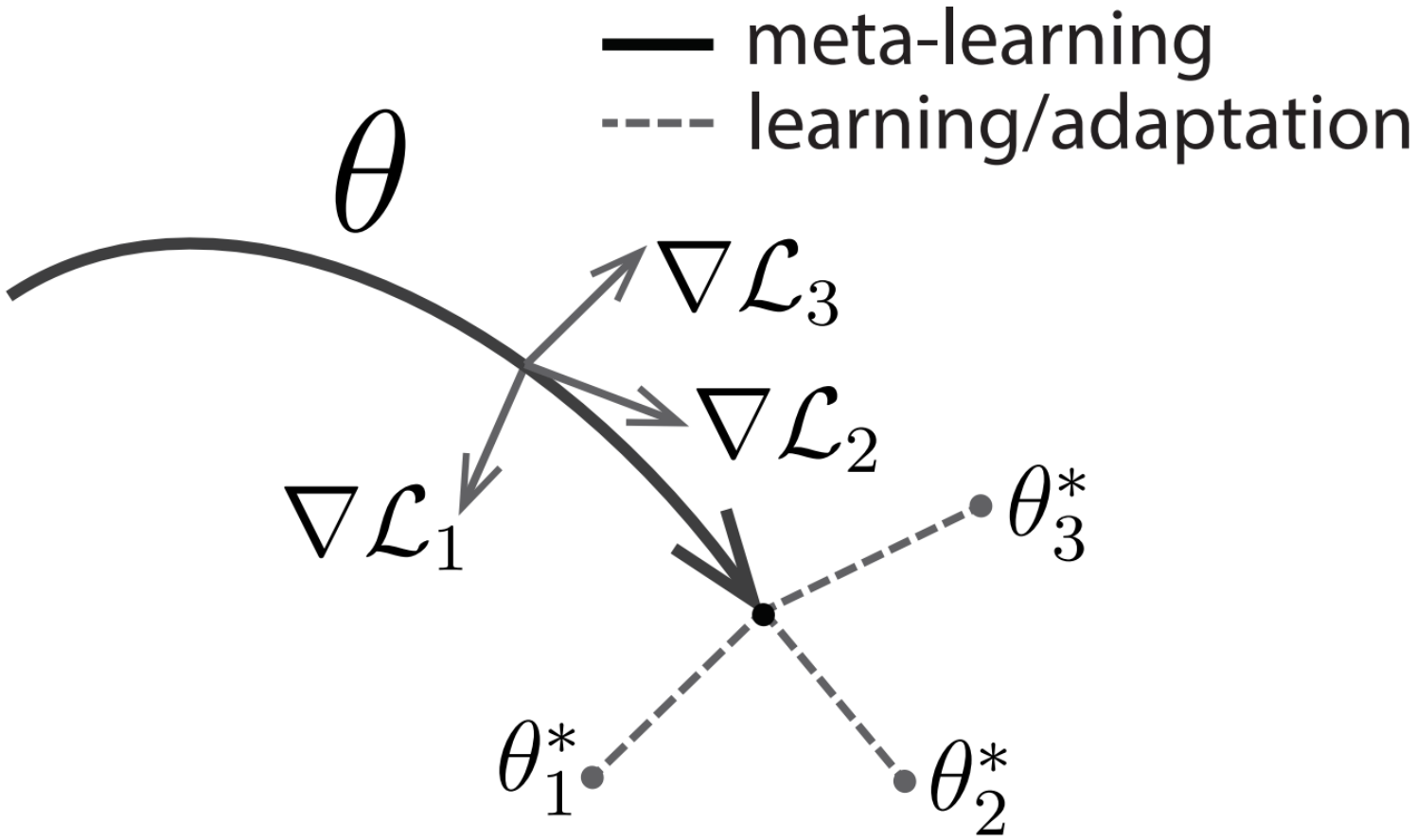
March 13, 2018

Motivation



In parameter space (μ, σ), each pair of distributions have the same distance.

Motivation



We often talk about the **parameter space** of a neural network, instead of the **functions** that those parameters represent.

Gradient Descent

The gradient descent update

$$\Delta\theta = -\alpha \nabla_\theta \mathcal{L}$$

is the solution to the following optimization problem:

$$\arg\min_{\Delta\theta} [\nabla_\theta \mathcal{L}(\theta) \cdot \Delta\theta]$$
$$s.t. \|\Delta\theta\| \leq \delta.$$

We are optimizing a linear approximation of $\mathcal{L}$ within a trust region defined by the Euclidean metric in parameter space.

Natural Gradient

Consider a family of density functions $\mathcal{F} : \theta \to p(\mathbf{z})$ where $\theta \in \mathbb{R}^n$:

$$p_\theta(\mathbf{z}) := \mathcal{F}(\theta)(\mathbf{z}).$$

We naturally obtain a notion of closeness among different values of θ:

$$d(\theta_1, \theta_2) := KL(p_{\theta_1}(\mathbf{z}) || p_{\theta_2}(\mathbf{z})).$$

Natural Gradient

$$d(\theta_1, \theta_2) := KL(p_{\theta_1}(\mathbf{z}) || p_{\theta_2}(\mathbf{z})).$$

Let's see how d behaves near a given θ.

$$\begin{aligned} KL(p_\theta(\mathbf{z}) || p_{\theta+\Delta\theta}(\mathbf{z})) &= \mathbb{E}_{\mathbf{z}}\left[\log p_\theta\right] - \mathbb{E}_{\mathbf{z}}\left[\log p_\theta\right] \\ &\quad -\mathbb{E}_{\mathbf{z}}\left[\nabla \log p_\theta\right] \Delta\theta + O(\Delta\theta^2) \\ &= O(\Delta\theta^2), \end{aligned}$$

so no useful information from first order approximation.

Natural Gradient

$$d(\theta_1, \theta_2) := KL(p_{\theta_1}(\mathbf{z}) || p_{\theta_2}(\mathbf{z})).$$

Let's see how d behaves near a given θ.

$$\begin{aligned} KL(p_\theta(\mathbf{z}) || p_{\theta+\Delta\theta}(\mathbf{z})) &= -\frac{1}{2}\Delta\theta^\top \mathbb{E}_{\mathbf{z}}\left[\nabla^2 \log p_\theta\right]\Delta\theta + O(\Delta\theta^3) \\ &\approx \frac{1}{2}\Delta\theta^\top F_\theta \Delta\theta, \end{aligned}$$

where

$$F_\theta = \mathbb{E}_{\mathbf{z}}\left[-\nabla^2 \log p_\theta\right] = \mathbb{E}_{\mathbf{z}}\left[\nabla \log p_\theta \nabla \log p_\theta^\top\right].$$

The Fisher Information Matrix is the expected negative Hessian of $\log p$.

Natural Gradient

We change the constraint of

$$\arg\min_{\Delta\theta} [\nabla_\theta \mathcal{L}(\theta) \cdot \Delta\theta]$$
$$s.t.\ \|\Delta\theta\| = const$$

to

$$s.t. KL(p_\theta(\mathbf{z}) || p_{\theta+\Delta\theta}(\mathbf{z})) = const.$$

Natural Gradient

We change the constraint of

$$\arg\min_{\Delta\theta} [\nabla_\theta \mathcal{L}(\theta) \cdot \Delta\theta]$$
$$s.t.\ \|\Delta\theta\| = const$$

to

$$s.t. KL(p_\theta(\mathbf{z}) || p_{\theta+\Delta\theta}(\mathbf{z})) = const.$$

Solution after using Lagrange multiplier

$$\nabla_\theta \mathcal{L}(\theta) \cdot \Delta\theta + \frac{1}{2}\lambda \Delta\theta^\top F_\theta \Delta\theta$$

is

$$\Delta\theta \propto F_\theta^{-1} \nabla_\theta \mathcal{L}$$

Natural Gradient

For η defined as

$$\eta - \eta_0 = F^{\frac{1}{2}}(\theta - \theta_0),$$

the Fisher ball is a unit circle. Thus in the parameter space of η, the natural gradient is the same as the gradient.

Second-order Optimization

$$\Delta\theta = \arg\min_{\delta} M(\delta), \qquad M(\delta) := \frac{1}{2}\delta^\top B\delta + \nabla h(\theta)^\top \delta + h(\theta)$$

The solution is $\Delta\theta = -B^{-1}\nabla h$.

Second-order Optimization

$$\Delta\theta = \arg\min_{\delta} M(\delta), \qquad M(\delta) := \frac{1}{2}\delta^\top B\delta + \nabla h(\theta)^\top \delta + h(\theta)$$

The solution is $\Delta\theta = -B^{-1}\nabla h$.

- $B = \beta\mathcal{I}$: Gradient Descent.
- $B = H(\theta)$: Newton's Method.

Newton's method assumes h is convex, and fails otherwise.

Second-order Optimization

The Generalized Gauss-Newton Matrix

Let our loss function be $L(f(x,\theta))$.

$$
\begin{aligned}
H_{ij} &= \frac{\partial}{\partial\theta_j}\frac{\partial L(f(x,\theta))}{\partial\theta_i} \\
&= \sum_{k=0}^{n}\frac{\partial}{\partial\theta_j}\left(\frac{\partial L\left(f(x,\theta)\right)}{\partial f_k(x,\theta)}\frac{\partial f_k(x,\theta)}{\partial\theta_i}\right) \\
&= \sum_{k=0}^{n}\left(\sum_{l=0}^{n}\frac{\partial L^2\left(f(x,\theta)\right)}{\partial f_k(x,\theta)\partial f_l(x,\theta)}\frac{\partial f_k(x,\theta)}{\partial\theta_j}\right)\frac{\partial f_k(x,\theta)}{\partial\theta_i} \\
&+ \sum_{k=0}^{n}\frac{\partial L\left(f(x,\theta)\right)}{\partial f_k(x,\theta)}\frac{\partial f_k(x,\theta)^2}{\partial\theta_j\partial\theta_i} \\
\therefore H &= J_f^\top H_L J_f + \sum_{k=0}^{n}\frac{\partial L}{\partial f_k}H_{f_k} \approx J_f^\top H_L J_f = G
\end{aligned}
$$

Second-order Optimization

Why G instead of H

- G is positive semidefinite, H is not.
- Let's expand H further, assuming f is a feedforward neural network:

$$\cdots \xrightarrow{W_i} s_i \xrightarrow{\phi} a_i \xrightarrow{W_{i+1}} s_{i+1} \cdots$$

$$H - G = \sum_{i=1}^{l} \sum_{j=1}^{m_i} (\nabla a_i L)\, J_{s_i}^{\top} H_{[\phi(s_i)]_j} J_{s_i}$$

  The remaining term is the sum of curvature terms coming from each intermediate activation. These curvature terms are subject to more frequent change.
- In ReLU networks, $H_{[\phi(s_i)]} = 0$ a.e..

F and G

Define the conditional distribution r to be

$$r(y|z) = r(y|f(x,\theta))$$

We have

$$\nabla_\theta \log p(y; x, \theta) = J_f^\top \nabla_z \log r(y|z)$$

$$\begin{aligned} F &= \mathbb{E}_x \left[\nabla \log p(y; x, \theta) \nabla \log p(y; x, \theta)^\top \right] \\ &= \mathbb{E}_x \left[J_f^\top \nabla_z \log r(y|z) \nabla_z \log r(y|z)^\top J_f \right] \\ &= \mathbb{E}_x \left[J_f^\top F_R J_f \right] \\ G &= \mathbb{E}_x \left[J_f^\top H_L J_f \right] \end{aligned}$$

$\therefore F = G$ when $F_R = H_L$.

F and G

We know that $F = G$ when $F_R = H_L$. When does this occur?
Let $L(y, z) = -\log r(y|z)$.

$$\begin{aligned} F_R &= -\mathbb{E}_{R_{y|f(x,\theta)}}\left[H_{\log r}\right] \\ H_L &= -\mathbb{E}_{(x,y)}\left[H_{\log r}\right] \end{aligned}$$

$\therefore F_R = H_L$ holds when $\mathbb{E}_{R_{y|f(x,\theta)}}\left[H_{\log r}\right] = \mathbb{E}_{(x,y)}\left[H_{\log r}\right]$

F and G

We know that $F = G$ when $F_R = H_L$. When does this occur?
Let $L(y, z) = -\log r(y|z)$.

$$\begin{aligned} F_R &= -\mathbb{E}_{R_{y|f(x,\theta)}}\left[H_{\log r}\right] \\ H_L &= -\mathbb{E}_{(x,y)}\left[H_{\log r}\right] \end{aligned}$$

$\therefore F_R = H_L$ holds when $\mathbb{E}_{R_{y|f(x,\theta)}}\left[H_{\log r}\right] = \mathbb{E}_{(x,y)}\left[H_{\log r}\right]$

This holds when $r(y|z)$ is an exponential family:

$$\log r(y|z) = z^\top T(y) - \log Z(z)$$

since in this case, the hessian $H_{\log r}$ does not depend on y.
Most commonly-used losses satisfy this property, including the MSE loss and cross-entropy loss.

Summary

- Roughly speaking, the natural gradient is the direction of steepest change of loss in function space.
- The natural gradient is invariant under reparameterization.
- For most neural networks of interest, natural gradient descent is identical to a second-order method (GGN).

References I

[1] Shun-Ichi Amari. "Natural Gradient Works Efficiently in Learning". In: *Neural Comput.* (1998).

[2] James Martens. "Deep learning via Hessian-free optimization". In: *Proceedings of the International Conference on Machine Learning (ICML)* (2010).

[3] James Martens. *New insights and perspectives on the natural gradient method*. *Preprint arXiv:1412.1193*. 2014.

[4] James Martens and Roger B. Grosse. "Optimizing Neural Networks with Kronecker-factored Approximate Curvature". In: *Proceedings of the International Conference on Machine Learning (ICML)* (2015).

[5] Hyeyoung Park, Shun-Ichi Amari, and Kenji Fukumizu. "Adaptive natural gradient learning algorithms for various stochastic models". In: *Neural Networks* (2000).

References II

[6] Razvan Pascanu and Yoshua Bengio. "Revisiting Natural Gradient for Deep Networks". In: *Proceedings of the International Conference on Learning Representations (ICLR)* (2014).

[7] Oriol Vinyals and Daniel Povey. "Krylov Subspace Descent for Deep Learning". In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)* (2012).

Thank You