Fast yet Simple Natural-Gradient Descent for Variational Inference in Complex Models

Mohammad Emtiyaz Khan
RIKEN Center for Advanced Intelligence Project
Tokyo, Japan
emtiyaz.khan@riken.jp

Didrik Nielsen
RIKEN Center for Advanced Intelligence Project
Tokyo, Japan
didrik.nielsen@riken.jp

***Abstract*—Bayesian inference plays an important role in advancing machine learning, but faces computational challenges when applied to complex models such as deep neural networks. Variational inference circumvents these challenge by formulating Bayesian inference as an optimization problem and solving it using gradient-based optimization. In this paper, we argue in favor of *natural-gradient* approaches which, unlike their *gradient*-based counterparts, can improve convergence by exploiting the information geometry of the solutions. We show how to derive fast yet simple natural-gradient updates by using a duality associated with exponential-family distributions. An attractive feature of these methods is that, by using natural-gradients, they are able to extract accurate local approximations for individual model components. We summarize recent results for Bayesian deep learning showing the superiority of natural-gradient approaches over their gradient counterparts.**

***Index Terms*—Bayesian inference, variational inference, natural gradients, stochastic gradients, information geometry, exponential-family distributions, nonconjugate models.**

I. Introduction

Modern machine-learning methods, such as deep learning, are capable of producing accurate predictions which has lead to their enormous recent success in fields, e.g., computer vision, speech recognition, and recommendation systems. However, this is not enough for other fields such as robotics and medical diagnostics where we also require an accurate estimate of confidence or uncertainty in the predictions. Bayesian inference provides such uncertainty measures by using the *posterior distribution* obtained using Bayes' rule. Unfortunately, this computation requires integrating over all possible values of the model parameters, which is infeasible for large complex models such as Bayesian neural networks.

Sampling methods such as Markov Chain Monte Carlo usually converge slowly when applied to such large problems. In contrast, approximate Bayesian methods such as variational inference (VI) can scale to large problems by obtaining approximations to the posterior distribution by using an optimization method, e.g., stochastic-gradient descent (SGD) methods [4], [6], [16]. These methods could provide reasonable approximations very quickly.

An issue in using SGD is that it ignores the information geometry of the posterior approximation (see Figure 1(a)). Recent approaches address this issue by using stochastic *natural-gradient* descent methods which exploit the Riemannian geometry of exponential-family approximations to improve the rate of convergence [7]–[9]. Unfortunately, these approaches only apply to a restricted class of models known as *conjugate* models, and do not work for nonconjugate models such as Bayesian neural networks.

This paper discusses some recent methods that generalize the use of natural gradients to such large and complex nonconjugate models. We show that, for exponential-family approximations, a duality between their natural and expectation parameters enables a simple natural-gradient update. The resulting updates are equivalent to a recently proposed method called Conjugate-computation Variational Inference (CVI) [10]. An attractive feature of the method is that it naturally obtains *local* exponential-family approximations for individual model components. We discuss the application of the CVI method to Bayesian neural networks and show some recent results from a recent work [11] demonstrating faster convergence of natural-gradient VI methods compared to gradient-based VI methods (see Figure 1(b)).

II. Problem Formulation

In this section, we discuss the problem of variational inference and show how SGD can be used to optimize it. SGD ignores the geometry of the posterior approximations, and we discuss how natural-gradient methods address this issue. We end the section by mentioning issues with existing natural-gradient methods for variational inference.

A. Variational Inference (VI)

We consider models[1] that take the following form:

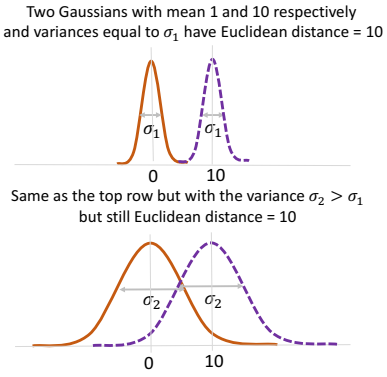
$$p(\mathcal{D}, \mathbf{z}) \propto \left[\prod_{i=1}^{N} p(\mathcal{D}_i|\mathbf{z})\right] p(\mathbf{z}), \text{ where } \mathcal{D} := \{\mathcal{D}_i\}_{i=1}^{N} \quad (1)$$

where p is a likelihood function which relates the model parameters $\mathbf{z}$ to the i'th data-example $\mathcal{D}_i$, and $p(\mathbf{z})$ is the prior distribution which we assume to be an exponential-family distribution [21],

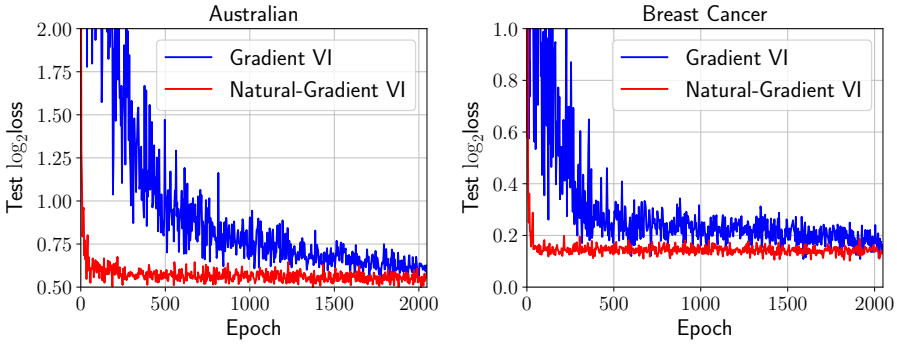
$$p(\mathbf{z}) \propto h(\mathbf{z}) \exp\left\{\boldsymbol{\phi}(\mathbf{z})^\top \boldsymbol{\eta}_0 - A(\boldsymbol{\eta}_0)\right\}, \text{ where } \boldsymbol{\eta}_0 \in \Omega \quad (2)$$

where $\boldsymbol{\phi}$ is a vector of sufficient statistics, $\boldsymbol{\eta}_0$ is the natural-parameter vector, and $A(\boldsymbol{\lambda})$ is the log-partition function. The

[1]Methods discussed in this paper apply to a more general class of models, e.g., the model class discussed in [10], but for clarity of presentation we focus on a restricted class.



(a) Gradient-based methods use the Euclidean distance which is a poor metric to measure distance between distributions. The bottom two distributions are almost identical while the top ones barely overlap, yet Euclidean distance is the same.



(b) Natural-gradient method could converge faster than gradient-based methods. We apply a Bayesian neural network on two datasets, namely the Australian dataset (shown in left) and the Breast Cancer dataset (shown in right). A lower value of the test $\log_2$loss is considered better. The natural-gradient method is the Variational Online Gauss-Newton (VOGN) method proposed in [11] while the gradient method is the Bayes-by-Backprop method proposed in [4]. The latter uses the Adam optimizer [12].

Fig. 1.

model parameter is a random vector here and sometimes is referred to as the *latent* vector.

Example: Consider Bayesian neural networks (BNN) [3] to model data $\mathcal{D}_i$ that contains input $\mathbf{x}_i \in \mathbb{R}^D$ and a scalar output y_i. The vector $\mathbf{z}$ is the vector of network weights. The likelihood $p(\mathcal{D}_i|\mathbf{z})$ could be an exponential-family distribution $p(y_i|f_{\mathbf{z}}(\mathbf{x}_i))$ whose expectation parameter $f_{\mathbf{z}}(\cdot)$ is a neural network parameterized by $\mathbf{z}$. We assume an isotropic Gaussian prior $p(\mathbf{z}) := \mathcal{N}(\mathbf{z}|0, \mathbf{I}/\tau)$ where τ is a scalar. Its natural parameters are $\boldsymbol{\eta}_0 := \{0, -\tau\mathbf{I}/2\}$. □

For such models, Bayesian approaches can estimate a measure of uncertainty by using the posterior distribution: $p(\mathbf{z}|\mathcal{D}) := p(\mathcal{D}|\mathbf{z})p(\mathbf{z})/p(\mathcal{D})$. This requires computation of the normalization constant $p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$ which is unfortunately difficult to compute in models such as Bayesian neural network. One source of difficulty is that the likelihood $\tilde{p}(\mathcal{D}_i|\mathbf{z})$ does not take the same form as the prior with respect to $\mathbf{z}$, or, in other words, the model is *nonconjugate* [5]. As a result, the product $p(\mathcal{D}|\mathbf{z})p(\mathbf{z})$ does not take a form with which $p(\mathcal{D})$ can be easily computed.

Variational inference (VI) simplifies the problem by approximating $p(\mathbf{z}|\mathcal{D})$ with a distribution $q(\mathbf{z})$ whose normalizing constant is relatively easier to compute. In models (1), a straightforward choice is to choose $q(\mathbf{z})$ to be of the same parametric[2] form as the prior $p(\mathbf{z})$ but with a different natural-parameter vector $\boldsymbol{\lambda}$, i.e., $q_\lambda(\mathbf{z}) := h(\mathbf{z}) \exp[\boldsymbol{\lambda}^\top \boldsymbol{\phi}(\mathbf{z}) - A(\boldsymbol{\lambda})]$. The parameter $\boldsymbol{\lambda}$ can be obtained by maximizing the variational objective which is also a lower bound to $p(\mathcal{D})$ [3],

$$\max_{\lambda\in\Omega} \mathcal{L}(\boldsymbol{\lambda}) := \mathbb{E}_{q_\lambda}\left[\log \frac{p(\mathbf{z})}{q_\lambda(\mathbf{z})}\right] + \sum_{i=1}^{N} \mathbb{E}_{q_\lambda}[\log p(\mathcal{D}_i|\mathbf{z})]. \quad (3)$$

[2]This restriction may not lead to a suboptimal approximation, e.g., in mean-field approximation in conjugate exponential-family models, the optimal form according to the variational objective turns out to be an exponential-family approximation [3].

where Ω is the set of valid variational parameters. Intuitively, the first term favors $q_\lambda(\mathbf{z})$ which is close to the prior $p(\mathbf{z})$ while the second term favors those that obtain high expected log-likelihood values. The variational objective has a very familiar form similar to many other *regularized* optimization problems in machine learning [3].

Example: In the BNN example, we can choose $q_\lambda(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{m}, \mathbf{V})$ where $\mathbf{m}$ is the mean and $\mathbf{V}$ is the covariance. The natural-parameter vector is $\boldsymbol{\lambda} := \{\mathbf{V}^{-1}\mathbf{m}, -\frac{1}{2}\mathbf{V}^{-1}\}$, and our goal in VI is to maximize $\mathcal{L}$ with respect to these parameters. □

B. VI with Gradient Descent

A straightforward approach to maximize $\mathcal{L}$ is to use a gradient-based method, e.g., the following stochastic-gradient descent (SGD) algorithm:

$$\boldsymbol{\lambda}_{t+1} = \boldsymbol{\lambda}_t + \rho_t \left[\widehat{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}_t)\right], \quad (4)$$

where t is the iteration number, ρ_t is a step size, and $\widehat{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}_t) := \widehat{\partial \mathcal{L}}/\partial \boldsymbol{\lambda}$ is a stochastic estimate of the derivative of $\mathcal{L}$ at $\boldsymbol{\lambda} = \boldsymbol{\lambda}_t$ (the 'hat' here indicates a stochastic estimate). Such stochastic gradients are easily computed using methods such as REINFORCE [22] and the reparameterization trick [13], [18]. This results in a simple but powerful approach which applies to many models and scales to large data.

Despite this, a direct application of SGD to optimize $\mathcal{L}(\boldsymbol{\lambda})$ is problematic because SGD ignores the *information geometry* of the distribution $q_\lambda(\mathbf{z})$. To see this, we can rewrite (4) as,

$$\boldsymbol{\lambda}_{t+1} = \arg\max_{\lambda}\ \boldsymbol{\lambda}^\top \left[\widehat{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}_t)\right] - \frac{1}{2\rho}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_t\|^2, \quad (5)$$

Equivalence can be established by taking the derivative and setting to 0. The equation (5) implies that SGD moves in the direction of the gradient while remaining close, in terms of the Euclidean distance, to the previous $\boldsymbol{\lambda}_t$. However, the Euclidean distance between natural parameters is not appropriate because

$\boldsymbol{\lambda}$ is the parameter of a distribution and the Euclidean distance is often a poor measure of dissimilarity between distributions. This is illustrated in Figure 1(a). The Euclidean distance used by SGD is therefore a poor candidate to measure distance in parameter space and a more informative measure which directly measure the distance between distributions might be more appropriate.

C. VI with Natural-Gradient Descent

The issue discussed above can be addressed by using natural-gradient methods that exploit the information geometry of q [1]. An exponential-family distribution induces a Riemannian manifold with a metric defined by the Fisher Information Matrix (FIM) [2], e.g. the FIM can be obtained as follows in the natural parameterization,

$$\mathbf{F}(\boldsymbol{\lambda}) := \mathbb{E}_{q_\lambda}\left[\nabla_\lambda \log q_\lambda(\mathbf{z})\ \nabla_\lambda \log q_\lambda(\mathbf{z})^\top\right] \tag{6}$$

Natural-gradient descent modifies the SGD step (5) by using the Riemannian metric instead of the Euclidean distance,

$$\max_{\lambda}\ \boldsymbol{\lambda}^\top\left[\widehat{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}_t)\right] - \frac{1}{2\alpha_t}(\boldsymbol{\lambda}-\boldsymbol{\lambda}_t)^\top \mathbf{F}(\boldsymbol{\lambda}_t)(\boldsymbol{\lambda}-\boldsymbol{\lambda}_t), \tag{7}$$

where $\alpha_t > 0$ is a scalar step size. This results in an update analogue to (4),

$$\boldsymbol{\lambda}_{t+1} = \boldsymbol{\lambda}_t + \alpha_t \left[\mathbf{F}(\boldsymbol{\lambda}_t)\right]^{-1} \widehat{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}_t), \tag{8}$$

where the stochastic gradient is scaled by the FIM. The scaled stochastic-gradient is referred to as the stochastic *natural gradient* defined as follows:

$$\widetilde{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}) := \left[\mathbf{F}(\boldsymbol{\lambda})\right]^{-1} \widehat{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}). \tag{9}$$

We use the notation $\tilde{\nabla}$ to differentiate the *natural* gradient as opposed to the standard gradient in Euclidean space denoted by ∇. In practice, the scaling, in a similar spirit to Newton's method, improves convergence and also simplifies step-size tuning.

Natural gradients are also naturally suited for VI in certain class of models. A recent work in [8] shows that for conjugate exponential-family models, natural-gradients with respect to the natural-parameterization take a very simple form. For example, consider the first term in (3) which consists of the ratio of two terms that are conjugate to each other. The natural-gradient then is equal to the difference in the natural parameter of the two terms:

$$\widetilde{\nabla}_\lambda \mathbb{E}_{q_\lambda}\left[\log \frac{p(\mathbf{z})}{q_\lambda(\mathbf{z})}\right] = \boldsymbol{\eta}_0 - \boldsymbol{\lambda} \tag{10}$$

The above natural gradient does not require computation of the FIM, which is surprising. It is natural to ask whether a similar expression is possible when the model contains nonconjugate terms? We show that it is possible to do so if we perform natural-gradient descent in the natural parameter space, but not if we do it in the space of expectation parameters.

III. Natural Gradients with Exponential Family

In this section, we show that natural gradient with respect to the natural parameters can be obtained by computing the *gradient* with respect to the expectation parameter. In the next section, we will show that this enables a simple natural-gradient update which does not require an explicit inversion of FIM.

We start by defining the expectation[3] parameter $\boldsymbol{\mu} \in \mathbb{R}^M$ of an exponential-family distribution q_λ as follows: $\boldsymbol{\mu}(\boldsymbol{\lambda}) := \mathbb{E}_{q_\lambda}\left[\boldsymbol{\phi}(\mathbf{z})\right]$, where we have expressed $\boldsymbol{\mu}$ as a function of $\boldsymbol{\lambda}$. Alternatively, $\boldsymbol{\mu}$ can be obtained from the natural parameters by simply differentiating the log-partition function, i.e., $\boldsymbol{\mu}(\boldsymbol{\lambda}) = \nabla A(\boldsymbol{\lambda})$. The mapping ∇A is one-to-one and onto (a bijection) iff the representation is minimal. Therefore, we can express $\mathcal{L}(\boldsymbol{\lambda})$ in terms of $\boldsymbol{\mu}$. We denote the new objective by $\mathcal{L}_*(\boldsymbol{\mu}) := \mathcal{L}(\boldsymbol{\lambda})$. We can now state our claim.

Theorem 1. *For an exponential-family in the minimal representation, the natural gradient with respect to $\boldsymbol{\lambda}$ is equal to the gradient with respect to $\boldsymbol{\mu}$, and vice versa, i.e.,*

$$\widetilde{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}) = \nabla_\mu \mathcal{L}_*(\boldsymbol{\mu}) \text{ and } \widetilde{\nabla}_\mu \mathcal{L}_*(\boldsymbol{\mu}) = \nabla_\lambda \mathcal{L}(\boldsymbol{\lambda}) \tag{11}$$

Proof: *Using chain rule, we can rewrite the derivative with respect to $\boldsymbol{\lambda}$ in terms of $\boldsymbol{\mu}$:*

$$\nabla_\lambda \mathcal{L}(\boldsymbol{\lambda}) = \left[\nabla_\lambda \boldsymbol{\mu}\right] \nabla_\mu \mathcal{L}_*(\boldsymbol{\mu}) = \left[\nabla^2_{\lambda\lambda} A(\boldsymbol{\lambda})\right] \nabla_\mu \mathcal{L}_*(\boldsymbol{\mu}) \tag{12}$$

It is well known that the second derivative of $A(\boldsymbol{\lambda})$ is equal to the FIM for exponential-family distribution, i.e., $\mathbf{F}(\boldsymbol{\lambda}) := \nabla^2_{\lambda\lambda} A(\boldsymbol{\lambda})$ [15]. This matrix is invertible when the representation is minimal. Therefore multiplying the above equation with inverse of $\mathbf{F}(\boldsymbol{\lambda})$ gives us the first equality. Since the FIM with respect to $\boldsymbol{\lambda}$ is inverse of the FIM with respect to $\boldsymbol{\mu}$ [15], the second equality is immediate. □

This result is a consequence of a relationship between $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$. The two vectors are related through the *Legendre transform* which is the following transformation $\boldsymbol{\mu} = \nabla A(\boldsymbol{\lambda})$. Since $A(\boldsymbol{\lambda})$ is a convex function, the space of $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ are both Riemannian manifolds which are also duals[4] of each other. An attractive property of this structure is that the FIM in one space is the inverse of the FIM in the other space. This enables us to compute natural gradient in one space using the gradient in the other, as shown in (11). This result is also discussed in an earlier work by Hensman et al. [7] in the context of conjugate models, although they do not explicitly mention the connection to duality.

The natural gradient $\widetilde{\nabla}_\lambda$ makes a better choice for conjugate models because ∇_μ assumes a simple form which does not require computation of the FIM. The $\widetilde{\nabla}_\mu$ unfortunately does not have this property. For example, $\widetilde{\nabla}_\mu$ for (10) requires computation of the FIM because it is equal to $\mathbf{F}(\boldsymbol{\lambda})(\boldsymbol{\eta}_0 - \boldsymbol{\lambda})$. This can be shown by using (11), (9) and (10). This result might be puzzling since natural gradients are supposed to

[3] Sometimes also called the mean or moment parameter.

[4] In information geometry, this is known as the *dually-flat Riemannian structure* [2].

be invariant to parameterization. In fact, the two gradients should point in the same direction, but their computation typically require different types of computations. In general, the two algorithm, obtained by using $\widetilde{\nabla}_\mu$ and $\widetilde{\nabla}_\lambda$ respectively, will follow different paths, even though they are expected to converge to similar solutions.

The recent work by [10] propose to use the gradients with respect to $\boldsymbol{\mu}$ to perform natural gradient with respect $\boldsymbol{\lambda}$. They arrive at this conclusion by using the equivalence of mirror descent and natural-gradient descent. Our discussion above complements their work by using the duality of the two spaces.

IV. Natural Gradients for Nonconjugate Models

In this section, we show that in some cases the natural gradient of the nonconjugate term can be easily computed by using ∇_μ. We also show that the resulting update takes a simple form.

We start with the expression for $\widetilde{\nabla}_\lambda \mathcal{L}$. Using (11) and (10), it is straightforward to write this expression:

$$\widetilde{\nabla}_\lambda \mathcal{L}(\boldsymbol{\lambda}) := \boldsymbol{\eta}_0 - \boldsymbol{\lambda} + \sum_{i=1}^{N} \widehat{\nabla}_\mu \, \mathbb{E}_q[\log p(\mathcal{D}_i|\mathbf{z})]|_{\mu=\mu(\lambda)}, \quad (13)$$

where we have expressed $\boldsymbol{\mu}$ as a function of $\boldsymbol{\lambda}$. For notational convenience, we will denote i'th term inside the summation by $\tilde{\mathbf{g}}_i(\boldsymbol{\lambda}) := \widehat{\nabla}_\mu \mathbb{E}_q[\log p(\mathcal{D}_i|\mathbf{z})]|_{\mu=\mu(\lambda)}$.

A *stochastic natural-gradient descent update* can be obtained by using the gradient of a randomly sampled data example $\mathcal{D}_i$ and multiplying it by N, as shown below:

$$\boldsymbol{\lambda}_{t+1} = (1-\alpha_t)\boldsymbol{\lambda}_t + \alpha_t \left[\boldsymbol{\eta}_0 + N\tilde{\mathbf{g}}_i(\boldsymbol{\lambda}_t)\right], \quad (14)$$

where the gradient is multiplied by N to obtain an unbiased stochastic gradient. This update is equivalent to the update obtained in [10] where it is referred to as Conjugate-computation variational inference (CVI). In [10], this is derived using a mirror-descent formulation, while we use the duality of the exponential family (Theorem 1).

Unlike the SGD update, the natural-gradient update (14) only computes gradients of the nonconjugate terms, thereby requires less computation. We now give an example which shows that $\tilde{\mathbf{g}}_i$ assumes a simple form and can be computed easily using automatic-gradient methods.

Example: For the BNNs example, $\tilde{\mathbf{g}}_i(\boldsymbol{\lambda})$ can be obtained by using backpropagated gradients $\mathbf{g}_i(\mathbf{z}) := \nabla_{\mathbf{z}} \log p(y_i|f_{\mathbf{z}}(\mathbf{x}_i))$ and Hessians $\mathbf{H}_i(\mathbf{z}) := \nabla^2_{\mathbf{zz}} \log p(y_i|f_{\mathbf{z}}(\mathbf{x}_i))$. For a Gaussian q_λ, there are two expectation parameters: $\boldsymbol{\mu}_1 := \mathbb{E}_{q_\lambda}(\mathbf{z}) = \mathbf{m}$ and $\boldsymbol{\mu}_2 := \mathbb{E}_{q_\lambda}(\mathbf{z}\mathbf{z}^T) = \mathbf{m}\mathbf{m}^\top + \mathbf{V}$. As shown in [11], we can write gradients as follows:

$$\begin{aligned}\nabla_{\mu_1} \mathbb{E}_{q_\lambda}[\log p(y_i|f_{\mathbf{z}}(\mathbf{x}_i))] &= \mathbb{E}_{q_\lambda}[\mathbf{g}_i(\mathbf{z})] - 2\mathbb{E}_{q_\lambda}[\mathbf{H}_i(\mathbf{z})]\mathbf{m} \\ \nabla_{\mu_2} \mathbb{E}_{q_\lambda}[\log p(y_i|f_{\mathbf{z}}(\mathbf{x}_i))] &= \mathbb{E}_{q_\lambda}[\mathbf{H}_i(\mathbf{z})]. \end{aligned} \quad (15)$$

If we approximate the expectations using a single Monte Carlo sample $\mathbf{z}_t \sim q_\lambda(\mathbf{z})$, we can write the update in (14) *as*

$$\mathbf{m}_{t+1} = \mathbf{m}_t - \alpha_t \mathbf{V}_{t+1}\left[\tau \mathbf{m}_t - N\mathbf{g}_i(\mathbf{z}_t)\right] \quad (16)$$

$$\mathbf{V}^{-1}_{t+1} = (1-\alpha_t)\mathbf{V}^{-1}_t + \alpha_t\left[\tau\mathbf{I} - N\mathbf{H}_i(\mathbf{z}_t)\right]. \quad (17)$$

These updates take a form similar to Newton's method. The covariance matrix $\mathbf{V}_t$ plays a similar role to the Hessian in Newton's method and scales the gradient in the update of $\mathbf{m}_t$. The matrix itself contains a moving average of the past Hessians. It is, however, not common to compute Hessians for deep models, but, as we discuss in Section VI, we can use another approximation to simplify this computation. With such an approximation, these updates can be implemented efficiently within existing deep learning code-bases as discussed in [11]. □

Similarly to the above example, it might be possible to employ automatic-gradient methods to compute natural gradients in many models. A recent work [20] explores this possibility. Another stochastic approximation method discussed in [19] is also useful. For simple models, such as generalized linear models, where we can directly derive the distribution of the local variables, we can locally compute the gradients. This is discussed in [10] for generalized linear models, Gaussian processes, and linear dynamical systems with nonlinear likelihoods.

V. Local Approximations with Natural Gradients

We now show that natural gradients not only result in simple updates, but they also give local exponential-family approximations of the nonconjugate terms. An attractive feature of these approximation is that the natural gradient of a nonconjugate likelihood is also the natural parameter of its local approximation.

We start by analyzing the optimality condition of $\mathcal{L}$. First, by setting (13) to zero, we note that a maximum $\boldsymbol{\lambda}_*$ of $\mathcal{L}$ satisfies the following[5]: $\boldsymbol{\lambda}_* = \boldsymbol{\eta}_0 + \sum_{i=1}^N \tilde{\mathbf{g}}_i(\boldsymbol{\lambda}_*)$. Then, multiplying by $\boldsymbol{\phi}(\mathbf{z})$, exponentiating the whole equation, and by using the definition (2) of the prior, we can rewrite the optimality condition as follows,

$$q(\mathbf{z}|\boldsymbol{\lambda}_*) \propto \left[\prod_{i=1}^{N} e^{\boldsymbol{\phi}(\mathbf{z})^\top \tilde{\mathbf{g}}_i(\boldsymbol{\lambda}_*)}\right] p(\mathbf{z}) \quad (18)$$

Comparing this update to the original model (1), we see that the nonconjugate likelihoods are replaced by *local exponential-family approximations* whose natural parameters are the *local natural-gradients* $\tilde{\mathbf{g}}_i(\boldsymbol{\lambda}_*)$. This type of local approximation is employed in Expectation Propagation (EP) [14]. In contrast, here they naturally emerge during a *global* step, i.e., during the optimization of the whole variational objective.

We denote the i'th local approximation at iteration t by $\tilde{q}_i^{(t)}$ and define it as follows,

$$p(\mathcal{D}_i|\mathbf{z}) \approx \tilde{q}_i^{(t)}(\mathbf{z}) \;\propto\; h(\mathbf{z})\; e^{\boldsymbol{\phi}(\boldsymbol{z})^\top \tilde{\boldsymbol{g}}_i(\boldsymbol{\lambda}_t)}, \quad (19)$$

We can then write the update (14) as an *approximate* Bayesian filter as shown below,

$$q_{\lambda_{t+1}}(\mathbf{z}) \propto \left[q_{\lambda_t}(\mathbf{z})\right]^{1-\alpha_t}\left[\left\{\tilde{q}_i^{(t)}(\mathbf{z})\right\}^N p(\mathbf{z})\right]^{\alpha_t}. \quad (20)$$

[5] We note that a similar optimality condition is used in [19] although the connection to natural gradients is not discussed.

This update replaces each likelihood term in the model (1) by the i'th likelihood term, which is why $\tilde{q}_i^{(t)}$ is raised to the power N. All distributions in the above update take the same exponential form as q_λ, and therefore the resulting computation can therefore be performed using *conjugate computations*, i.e., by simply adding their natural parameters. This algorithm is referred to as Conjugate-computation VI (CVI) in [10].

Finally, if the parameters $\boldsymbol{\eta}_0$ of the prior distribution do not change with iterations, then we can further simplify the updates by pulling $p(\mathbf{z})$ out of the iterations and expressing the local parameters $\tilde{\boldsymbol{\lambda}}_i$ as a recursion as shown below,

$$q(\mathbf{z}|\boldsymbol{\lambda}_{t+1}) \propto \left[\prod_{i=1}^{N} e^{\boldsymbol{\phi}(\mathbf{z})^\top \tilde{\boldsymbol{\lambda}}_i^{(t)}}\right] p(\mathbf{z}) \tag{21}$$

$$\text{where } \tilde{\boldsymbol{\lambda}}_i^{(t)} = (1-\alpha_t)\tilde{\boldsymbol{\lambda}}_i^{(t-1)} + \alpha_t \delta_{i,t}\left[N\tilde{\mathbf{g}}_i(\boldsymbol{\lambda}_t)\right],\ \forall i$$

where $\delta_{i,t} = 1$ if i'th data point is selected in the t'th iteration. The natural-parameter $\tilde{\boldsymbol{\lambda}}_i$ is similar to the so-called *site parameters* in EP [17]. The parameters $\tilde{\boldsymbol{\lambda}}_i^{(t)}$ converge to the natural parameters $\tilde{\mathbf{g}}_i(\boldsymbol{\lambda}_*)$ for the optimal local approximations.

VI. Results on Bayesian Neural Networks

In this section, we compare an approximate natural-gradient VI method with a gradient-based VI method. The natural-gradient method employs two approximations to the update (16)-(17). The first approximation is to use a diagonal covariance matrix which enables a fast computation when dimensionality of $\mathbf{z}$ is large. The second approximation is to use a *generalized Gauss-Newton* approximation for the Hessian. This avoids the need to compute second-order derivatives making the implementation easier. The resulting method is called *Variational Online Gauss-Newton (VOGN)* [11].

Figure 1(b) compares VOGN with a gradient-based approach called Bayes by Backprop [4]. The latter optimizes $\mathcal{L}$ using the Adam optimizer. The results are obtained using a neural network with single-hidden layer of 64 hidden units and ReLU activations. A prior precision of $\tau = 1$, a minibatch size of 128 and 16 Monte-Carlo samples are used for all runs. The two figures show results on the following two datasets: 'Australian' ($N = 690$ and $D = 14$) and 'Breast Cancer' ($N = 569$ and $D = 10$) datasets. We show $\log_2$loss vs epochs, where a lower value indicates a better performance. We clearly see that the natural-gradient method is much faster than the gradient-based method. [11] discusses more such examples clearly showing the advantages obtained with natural-gradient approaches.

VII. Conclusions

In this paper, we discuss methods for natural-gradient descent in variational inference. Unlike gradient-based approaches, natural-gradient methods exploit the information geometry of the solution and can converge quickly. We review a few recent works and provide new insights using the duality associated with exponential-family approximations. We discuss an attractive property of the natural-gradient to obtain local conjugate approximations for individual model components. Finally, we showed some illustrative examples where these methods have been applied to perform Bayesian deep learning.

Acknowledgment

We would like to thank the following people at RIKEN, AIP for discussions and feedback: Aaron Mishkin, Frederik Kunstner, Voot Tangkaratt, and Wu Lin.

References

[1] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[2] Shun-ichi Amari. *Information geometry and its applications*. Springer, 2016.

[3] Christopher M Bishop. Pattern recognition. *Machine Learning*, 128:1–58, 2006.

[4] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

[5] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*, volume 2. Chapman & Hall/CRC Boca Raton, FL, USA, 2014.

[6] Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, pages 2348–2356, 2011.

[7] James Hensman, Magnus Rattray, and Neil D Lawrence. Fast variational inference in the conjugate exponential family. In *Advances in neural information processing systems*, pages 2888–2896, 2012.

[8] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[9] Antti Honkela and Harri Valpola. Unsupervised variational Bayesian learning of nonlinear models. In *Advances in Neural Information Processing Systems*, pages 593–600, 2004.

[10] Mohammad Emtiyaz Khan and Wu Lin. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In *International conference on Artificial Intelligence and Statistics*, 2017.

[11] Mohammad Emtiyaz Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable Bayesian deep learning by weight-pertubation in adam. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[12] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[13] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[14] T. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2001.

[15] Frank Nielsen and Vincent Garcia. Statistical exponential families: A digest with flash cards. *arXiv preprint arXiv:0911.4863*, 2009.

[16] Rajesh Ranganath, Sean Gerrish, and David M Blei. Black box variational inference. In *International conference on Artificial Intelligence and Statistics*, pages 814–822, 2014.

[17] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[18] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[19] Tim Salimans, David A Knowles, et al. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):837–882, 2013.

[20] Hugh Salimbeni, Stefanos Eleftheriadis, and James Hensman. Natural gradients in practice: Non-conjugate variational inference in gaussian process models. *arXiv preprint arXiv:1803.09151*, 2018.

[21] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1–2:1–305, 2008.

[22] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.