Momentum for Variational Learning at Scale

Anonymous authors
Paper under double-blind review

Abstract

Momentum is well-studied for deep learning, but the same cannot be said for variational learning at scale. Here, we revisit the current practices of momentum for variational learning and propose a new momentum where an exponential moving-average of the *site* functions is maintained. We show that this site-momentum not only generalizes several existing (non-Bayesian) momentum approaches for deep learning (such as MARS and SGDHess) but also speeds up variational learning at scale. For instance, for training Llama 3 (1.3B) on 100B tokens from FineWeb, site-momentum yields a lower perplexity of up to 0.76 throughout compared to the state-of-the-art method. Overall, our work provides new theoretical connections as well as practical improvements for variational learning with momentum.

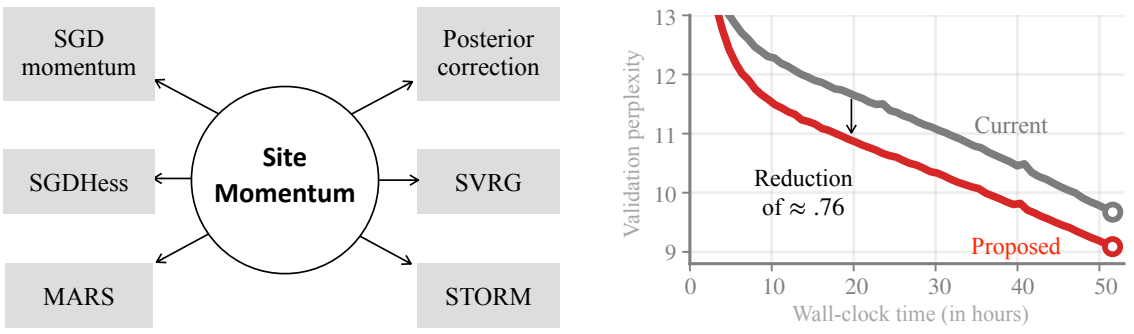
1 Introduction

Momentum is a popular technique in deep learning. Originally proposed by Polyak (1964), it was already used by Rumelhart et al. (1986) in the early days to train neural networks via stochastic gradient descent. Its importance was also quickly recognized when the new wave of deep learning arrived (Sutskever et al., 2013). Now, momentum is an integral part of deep-learning optimizers, of which most employ some form of momentum. More recently, it has also been used for variance reduction (Cutkosky & Orabona, 2019; Tran & Cutkosky, 2022; Yuan et al., 2025).

Here, we are concerned with the state of momentum for variational-Bayesian methods. In recent years, variational training has shown promising results for deep learning (Shen et al., 2024; Minut et al., 2026; Cong et al., 2025; Nishida et al., 2026; Daheim et al., 2026), but momentum for such methods is still not extensively studied. Variational learning uses a fundamentally different training objective than the standard empirical-risk minimization; the goal is to learn a posterior distribution over network parameters, not just a point estimate (Graves, 2011; Blundell et al., 2015; Osawa et al., 2019). Yet, the most common practice to incorporate momentum is to simply use the standard Adam-like momentum that was designed for non-Bayesian settings to learn point estimates. It remains unclear whether this is the most effective approach or if there are better alternatives.

In this paper, we revisit existing momentum methods for variational learning and propose new ways to improve them. The current practice is to maintain an exponential moving-average (EMA) of stochastic gradients of the *entire* objective. Instead, we propose to accumulate mini-batch gradients of the loss functions, leaving out the gradients of the entropy in the KL-term. When applied to the Bayesian learning rule of Khan & Rue (2023), this strategy can be expressed compactly via an EMA of the loss-surrogates called *site* functions. We call this EMA 'site-momentum' and show that it generalizes several (non-Bayesian) momentum methods for variance reduction (Fig. 1a), including MARS (Yuan et al., 2025) and SGDHess (Tran & Cutkosky, 2022). It also provides an efficient way to implement the posterior-correction method of Khan (2025); Daheim et al. (2026).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) (b)

Figure 1: Panel (a): We propose site-momentum for variational learning and connect it to several existing momentum methods for deep learning. Panel (b): Site-momentum (red) yields a lower perplexity compared to the current implementation of EVON for training Llama 3 (1.3B) training.

These results enable us to effortlessly derive new momentum by simply changing the form of the posterior distribution. For example, by employing Gaussian posteriors with diagonal and block-diagonal covariances, we derive two preconditioned MARS variants that respectively use a diagonal and block-diagonal preconditioner. The two variants can be easily implemented by changing a few lines of code in the existing variational-learning algorithms IVON (Shen et al., 2024) and EVON (Minut et al., 2026), respectively. Deriving such preconditioned variants is traditionally a cumbersome and difficult process. Site-momentum provides a new way to address this.

Our experiments confirm that the MARS variants of site-momentum significantly speed up variational training of large language models. We perform a range of experiments on nanoGPT and Llama (Touvron et al., 2023). Our largest experiment is on Llama 3 with 1.3B parameters trained on 100B tokens by using 32 GPUs in parallel with a batch size of around 1M tokens. For this experiment, site-momentum yields a lower perplexity compared to EVON. A drop in perplexity score of around 0.76 is observed throughout training (Fig. 1b). To the best of our knowledge, variational learning methods have never been shown to succeed at pretraining at this scale; this is the first time. In general, we observe larger gains at larger scale and longer runs. Gains are also higher for EVON compared to IVON clearly demonstrating the advantage of more flexible posteriors and second-order momentum. Final performance is also improved and gradient norm spikes are reduced.

2 Momentum for Deep Learning

Given a mini-batch gradient $\widehat{\mathbf{g}}_t$ of the loss $\ell(\boldsymbol{\theta})$ at iteration t, momentum takes the following form,

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \alpha\widehat{\mathbf{g}}_t + \beta(\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1}). \tag{1}$$

Here, α is the learning rate and β the momentum parameter. For deep learning, this update can be extremely effective (Sutskever et al., 2013). This is partly due to its stabilization property. The update is essentially equivalent to a gradient-descent step with accumulated gradients where past gradients are downweighted by β raised to powers based on their age; see Bertsekas (2011, Eq. 1.9). Due to its effectiveness, momentum has been extensively studied and used. A popular way to apply momentum is to use SGD with an exponential moving average of the gradients,

$$\begin{aligned} \bar{\mathbf{g}}_{t+1} &\leftarrow \beta_1\bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t, \\ \boldsymbol{\theta}_{t+1} &\leftarrow \boldsymbol{\theta}_t - \alpha\bar{\mathbf{g}}_{t+1}. \end{aligned} \tag{2}$$

Momentum is also employed in adaptive learning-rate methods like Adam (Kingma & Ba, 2015), and can be used for variance reduction, for instance, to implement SVRG (Johnson & Zhang, 2013). For instance, STORM (Cutkosky & Orabona, 2019) employs the following update,

$$\bar{\mathbf{g}}_{t+1} \leftarrow \beta_1 \bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t + \beta_1(\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t|t-1}), \tag{3}$$

where $\widehat{\mathbf{g}}_{t|t-1}$ is the gradient over the mini-batch sampled at iteration t but evaluated at $\boldsymbol{\theta}_{t-1}$. It is argued that by setting β_1 close to one, we can down-weight the noise in $\widehat{\mathbf{g}}_t$, while the noise in the gradient correction $\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t|t-1}$ is already expected to be tolerable. Motivated by a recent SVRG variant by Yin et al. (2025), Yuan et al. (2025) propose a variant of STORM called MARS. The method uses a few heuristics: it includes a coefficient γ in the last term and reuses $\widehat{\mathbf{g}}_{t-1}$ for $\widehat{\mathbf{g}}_{t|t-1}$ to save compute. The method also proposes new preconditioned variants of MARS. Another similar extension is by Tran & Cutkosky (2022). Recently, Semenov et al. (2025) show that MARS does not always yield good results for smaller models but is more effective for larger ones.

2.1 Momentum for Variational Learning

In this paper, we are concerned with the state of momentum techniques for variational learning. Despite its recent success at scale, little work has been done to improve the effectiveness of momentum. Existing approaches build upon Polyak's approach of Eq. 1 but end up approximating it to fit an Adam-like scheme (such as Eq. 2). We will first describe the basic setup of variational training with momentum for deep learning and highlight its main issues.

The goal of variational (Bayesian) learning is to optimize the following objective over a posterior density $q(\boldsymbol{\theta})$ over $\boldsymbol{\theta}$ in a candidate set $\mathcal{Q}$ (such as the set of Gaussian densities),

$$q_* = \arg\min_{q\in\mathcal{Q}} \; \kappa\, \mathbb{E}_q[\ell] + \mathbb{D}_{\mathrm{KL}}[q \,\|\, \pi_0]. \tag{4}$$

The first term is the expectation of the loss $\ell(\boldsymbol{\theta})$ under $q(\boldsymbol{\theta})$ which we will denote by $\mathcal{L}(q)$; $\kappa > 0$ is a scaling term. The second term is the regularizer set to the Kullback-Leibler divergence (KLD) to a prior density π_0. With this objective, variational training aims to find a posterior density such that samples $\boldsymbol{\theta} \sim q$ incur low loss while also ensuring the sample diversity to keep q closer to π_0.

Historically, efforts to scale variational training lagged behind the standard deep-learning approaches, but recent works have obtained competitive results by using the natural-gradient descent (NGD) approach of Khan & Rue (2023). They propose an algorithm called the Bayesian Learning Rule (BLR) to find solutions q that take an exponential-family (EF) form:

$$q(\boldsymbol{\theta}) \propto h(\boldsymbol{\theta}) \exp(\boldsymbol{\lambda}^\top \mathbf{T}(\boldsymbol{\theta})), \tag{5}$$

where $\boldsymbol{\lambda}$, $\mathbf{T}(\boldsymbol{\theta})$, and $h(\boldsymbol{\theta})$ denote the natural parameters, sufficient statistics, and the base measure, respectively. This family contains many well-known forms, for example, a Gaussian $\mathcal{N}(\boldsymbol{\theta} \mid \mathbf{m}, \mathbf{I})$ has EF form where $\boldsymbol{\lambda} = \mathbf{m}$, $\mathbf{T}(\boldsymbol{\theta}) = \boldsymbol{\theta}$, and $h(\boldsymbol{\theta}) = \exp(-\frac{1}{2}\boldsymbol{\theta}^\top\boldsymbol{\theta})$.

Given such a family, the BLR performs NGD in the space of $\boldsymbol{\lambda}$. As discussed in Khan & Rue (2023, App. F), the momentum-based BLR uses the following update (assume $\kappa = 1$ for now),

$$\boldsymbol{\lambda}_{t+1} \leftarrow \boldsymbol{\lambda}_t - \alpha(\tilde{\mathbf{g}}_t + \boldsymbol{\lambda}_t) + \beta(\boldsymbol{\lambda}_t - \boldsymbol{\lambda}_{t-1}). \tag{6}$$

The update is analogous to Eq. 1 but uses stochastic natural-gradients $\tilde{\mathbf{g}}_t = \mathbf{F}(\boldsymbol{\lambda}_t)^{-1}\widehat{\nabla}\mathcal{L}(q_t)$ defined using the Fisher $\mathbf{F}(\boldsymbol{\lambda}_t)$ of the density q_t parameterized by the natural parameter $\boldsymbol{\lambda}_t$. The stochastic gradient of the expected loss $\tilde{\mathbf{g}}_t$ also includes a contribution from the prior π_0. A minor difference is the $\boldsymbol{\lambda}_t$ in the second term which arises from the natural-gradient of the KLD in Eq. 4. This update takes a Bayes-like form. By first substituting $\boldsymbol{\lambda}_{t+1}$ in Eq. 5 and then simplifying we get:

$$q_{t+1} \leftarrow q_t^{1-\alpha} \exp\left(-\hat{\ell}_t\right)^{\alpha} \left(\frac{q_t}{q_{t-1}}\right)^{\beta}. \tag{7}$$

Here, we denote by $\hat{\ell}_t(\boldsymbol{\theta}) = \tilde{\mathbf{g}}_t^\top \mathbf{T}(\boldsymbol{\theta})$ the surrogate of the loss, also known as the 'site' function. A site plays a role similar to a linearized-loss surrogate $\widehat{\mathbf{g}}_t^\top \boldsymbol{\theta}$ used in SGD. The last term containing the ratio q_t/q_{t-1} is the momentum term. The operation '$\leftarrow$' here assumes normalization of the right-hand-side quantity to get a valid density q_{t+1} that integrates to 1 over its support.

Practical ways to implement BLR have been studied extensively, but little work has been done on momentum. The most-studied case is for Gaussian $q = \mathcal{N}(\boldsymbol{\theta} \mid \mathbf{m}, \text{diag}(\boldsymbol{\sigma})^2)$ with vector of standard deviation $\boldsymbol{\sigma} > 0$. For this posterior form, Khan et al. (2018) showed that the BLR can be conveniently implemented via Adam's codebase using a line-by-line correspondence to RMSProp. However, to bring momentum to the same form as that of Adam, they use an approximation by replacing instances of $\boldsymbol{\sigma}_{t-1}$ by $\boldsymbol{\sigma}_t$; see Khan et al. (2018, Eq. 79-80). In practice, this did not hurt performance and has later been shown to work well at large scale using IVON (Shen et al., 2024) and EVON (Minut et al., 2026) which extends these ideas to block-diagonal covariances. Lin et al. (2023) and Kiral et al. (2023) propose momentum from the viewpoint of Riemannian optimization and Lie groups, but these theoretical ideas have not been adopted in practice.

Despite these successes, it remains unclear if the momentum strategy used in IVON and EVON indeed is the best way or whether there is any room for improvement. In addition, it remains unclear whether extensions like STORM and MARS for variance reduction via momentum can also be connected to the BLR's momentum. A relevant work in this direction is by Daheim et al. (2026) who generalize SVRG by applying 'posterior correction' to the BLR (Khan, 2025). This raises the possibility that variance-reduction can also be achieved for the BLR via a momentum strategy. Next, we will propose an extension of the momentum-based BLR via *site-momentum* which not only yields a better new momentum for variational learning but also seamlessly generalizes STORM and MARS, giving us a new way to implement variance-reduction in the BLR through the momentum.

3 A New Site-Based Momentum For Variational Learning

We will now present two new momentum strategies for variational learning, arguing that straightforward application of Polyak's method as in Eq. 6 is not ideal because it accumulates the natural gradient of the whole variational objective rather than just those of the loss terms. We fix this by using a new site-momentum where only the natural gradients of the expected-loss are accumulated, which generalizes the approaches of Tran & Cutkosky (2022). Next, we will propose a generalization of MARS by using a posterior-correction approach (Khan, 2025; Daheim et al., 2026) which will also shed light on the heuristic used to set learning rates in MARS. Finally, we will discuss how our proposals connect to SARAH and recent Bayesian primal-dual algorithms (Möllenhoff et al., 2026).

As discussed earlier, Polyak's method in Eq. 1 accumulates the stochastic gradients $\widehat{\mathbf{g}}_t$. This is also clear from the form in Eq. 2 which expresses the update as an EMA of the gradients. In contrast, we can show that the momentum in Eq. 6 does not accumulate the natural gradients $\tilde{\mathbf{g}}_t$. Instead, it accumulates $\tilde{\mathbf{g}}_t + \boldsymbol{\lambda}_t$. This becomes clearer when writing it in an EMA form analogously to Eq. 2:

$$\begin{aligned} \bar{\boldsymbol{\lambda}}_{t+1} &\leftarrow \beta_1 \bar{\boldsymbol{\lambda}}_t + (1-\beta_1)(\tilde{\mathbf{g}}_t + \boldsymbol{\lambda}_t) \\ \boldsymbol{\lambda}_{t+1} &\leftarrow \boldsymbol{\lambda}_t - \alpha \bar{\boldsymbol{\lambda}}_{t+1}. \end{aligned} \tag{8}$$

A proof is in Sec. A. In fact, momentum is not actually implemented this way in practice. This is because the approximations used in Khan et al. (2018), which are later adopted by IVON and EVON, ultimately reduce this momentum to a gradient EMA which has a form similar to Eq. 2. For a few examples of this implementation, see pseudo-codes of Vadam (Khan et al., 2018, line 5, Fig. 1), VOGN (Osawa et al., 2019, line 17, Alg. 1), iBLR (Lin et al., 2020, line 5, Fig. 1), IVON (Shen et al., 2024, line 4, Alg. 1), and EVON (Minut et al., 2026, line 5, Alg. 2). All such implementations ultimately use the EMA of the gradients which is different from the BLR momentum in Eq. 8.

3.1 Momentum via the Site EMA

Our first proposal is to maintain an EMA of the natural gradients $\tilde{\mathbf{g}}_t$, without the $\boldsymbol{\lambda}_t$ term. This can also be written in terms of the sites because they are defined using the natural gradients. The two equivalent forms of our proposal are shown below with updates in $\boldsymbol{\lambda}$-space and q-space respectively,

$$\begin{aligned} \bar{\boldsymbol{\lambda}}_{t+1} &\leftarrow \beta_1 \bar{\boldsymbol{\lambda}}_t + (1-\beta_1)\tilde{\mathbf{g}}_t \qquad &\Longrightarrow \qquad \bar{\ell}_{t+1} &\leftarrow \beta_1 \bar{\ell}_t + (1-\beta_1)\hat{\ell}_t \\ \boldsymbol{\lambda}_{t+1} &\leftarrow (1-\alpha)\boldsymbol{\lambda}_t - \alpha \bar{\boldsymbol{\lambda}}_{t+1} \qquad &\Longrightarrow \qquad q_{t+1} &\leftarrow q_t^{1-\alpha} \exp\left(-\bar{\ell}_{t+1}\right)^{\alpha}, \end{aligned} \tag{9}$$

where on the right side we denote the EMA of the sites by $\bar{\ell}_t = \bar{\boldsymbol{\lambda}}_t^\top \mathbf{T}(\boldsymbol{\theta})$. This momentum scheme directly maintains the natural gradients of the expected-loss and uses it in the BLR. Eq. 9 assumes a constant base measure, and a general form for nonconstant base measures is given in Sec. B.1.

Theorem 1. *For the isotropic Gaussian family of the form $q(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \,|\, \mathbf{m}, \mathbf{I})$, Eq. 9 reduces to a form equivalent to the update of Eq. 2.*

A more rigorous statement and a proof are given in Sec. B.1. More precisely, we can implement the BLR update for this case via the SGD momentum by replacing $\widehat{\mathbf{g}}_t$ with a mini-batch gradient evaluated at a sample $\boldsymbol{\theta} \sim q_t$. SGD with momentum is recovered by replacing the sample by the mean $\mathbf{m}$. This way, our new proposal can be seen as an extension of the existing SGD momentum by incorporating sampling.

Our variational approach offers another way to generalize SGD momentum by using different posterior forms. We will now demonstrate this for a full-covariance Gaussian family.

Theorem 2. *For the Gaussian family with $q(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \,|\, \mathbf{m}, \boldsymbol{\Sigma})$, the site EMA in Eq. 9 can be implemented as follows:*

$$\begin{aligned} \bar{\mathbf{g}}_{t+1} &\leftarrow \beta_1 \bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t + \beta_1 \bar{\mathbf{H}}_t(\mathbf{m}_t - \mathbf{m}_{t-1}) \\ \bar{\mathbf{H}}_{t+1} &\leftarrow \beta_1 \bar{\mathbf{H}}_t + (1-\beta_1)\widehat{\mathbf{H}}_t. \end{aligned} \tag{10}$$

$\widehat{\mathbf{g}}_t$ and $\widehat{\mathbf{H}}_t$ are stochastic mini-batch gradients and Hessians evaluated at samples $\boldsymbol{\theta} \sim q_t$.

We call this a second-order momentum because we now maintain an EMA of both the gradients and Hessians. Remarkably, this has a strikingly similar form to the proposal of Tran & Cutkosky (2022) who call their approach SGD with Hessian-corrected momentum or SGDHess. The following Lemma states the result. Proofs are given in Sec. B.2.

Lemma 1. *The site momentum given in Eq. 10 reduces to the SGDHess approach of Tran & Cutkosky (2022) when $\widehat{\mathbf{g}}_t$ and $\widehat{\mathbf{H}}_t$ are computed at the mean $\mathbf{m}_t$ of q_t (instead of a sample from it) and we replace the accumulated Hessian $\bar{\mathbf{H}}_t$ with the Hessian $\widehat{\mathbf{H}}_t$ at the current point.*

SGDHess is derived by following the approach of STORM (Cutkosky & Orabona, 2019). Essentially, the gradient correction in Eq. 3 can be approximated as follows by using Taylor's expansion,

$$\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t|t-1} \approx \widehat{\mathbf{H}}_t(\mathbf{m}_t - \mathbf{m}_{t-1}) \tag{11}$$

In contrast, we show that site momentum naturally recovers it by using the full-Gaussian family providing an alternative justification and derivation for such ideas.

SGDHess (Tran & Cutkosky, 2022) is not empirically tested on the latest transformer-based language models. We will show empirical results showing that the approach can indeed be effective and can improve performance for language model training. We can efficiently implement second-order momentum within the IVON and EVON frameworks as these methods avoid an explicit computation of full Hessian-vector products which cause SGDHess to have a significant overhead. An algorithm using second-order momentum (SOM) in IVON is in Alg. 1 (full code in Sec. D).

3.2 Momentum via the Variance-Reduced Site EMA

We will now propose a generalization of STORM and MARS using our site-momentum approach. Motivated by this, we will build upon the variance reduction approach of Daheim et al. (2026). This approach generalizes the Stochastic Variance-Reduced Gradient (SVRG) method via the posterior-correction (PoCo) framework of Khan (2025). The key idea is to accumulate large-batch gradients in an outer loop and then use them to reduce the variance of inner BLR steps via a correction step. Specifically, given an outer loop distribution q^{out}, we perform multiple inner-loop BLR steps,

$$q_{t+1}^{\text{in}} \leftarrow \left(q_t^{\text{in}}\right)^{1-\eta} \left(q^{\text{out}}\right)^{\eta} \exp\left[-\frac{N}{B}\left(\hat{\ell}_{t|\text{in}} - \hat{\ell}_{t|\text{out}}\right)\right]^{\eta}. \tag{12}$$

N and B are the respective sizes of the data and mini-batch. Intuitively, we include q^{out} to stabilize the updates and then correct the inner-loop sites $\hat{\ell}_{t|\text{in}}$ evaluated at q^{in} by the outer-loop sites $\hat{\ell}_{t|\text{out}}$ evaluated at q^{out} to avoid double-counting the sites already in q^{out}. Unfortunately, just like the original SVRG, PoCo requires extra gradient computations in the outer loop and again in the inner loop which triples the cost, thus ultimately yielding no effective gain in runtime, but also no harm.

We will now show an efficient implementation of variance reduction via site-momentum. Our main idea is to use q_t in Eq. 9 as a replacement of q^{out} and to use the site EMA $\bar{\ell}_t$ as a replacement of $\hat{\ell}_{t|\text{out}}$. This avoids all extra computation by reusing these quantities:

$$\begin{aligned}
\bar{\ell}_{t+1} &\leftarrow \beta_1 \bar{\ell}_t + (1-\beta_1)\hat{\ell}_t \\
q_{t+1} &\leftarrow \left(q_t^{\text{in}}\right)^{1-\alpha} \exp\left(-\bar{\ell}_{t+1}\right)^{\alpha} \\
q_{t+1}^{\text{in}} &\leftarrow \left(q_t^{\text{in}}\right)^{1-\eta} \left(q_{t+1}\right)^{\eta} \exp\left[-\tau\left(\hat{\ell}_t - \bar{\ell}_t\right)\right]^{\eta}.
\end{aligned} \tag{13}$$

A major change compared to site-momentum in Eq. 9 is that the BLR step in the second line is taken at the latest q_t^{in}. Also, we have replaced N/B in Eq. 12 by a hyperparameter τ. This is because it is difficult to determine the contribution of sites in the site EMA. It is clear that these updates do not require any additional gradient computations by reusing the current $\hat{\ell}_t$.

We call the above approach Variance-Reduced Momentum (VRM). Note that, for $\tau = 0$ and $\eta = 1$, we go back to site-momentum. The following theorem additionally shows a specific choice of learning rates for which we can recover MARS-like updates.

Theorem 3. *Define $\bar{s}_{t+1} = \bar{\ell}_{t+1} + \beta_1\gamma(\hat{\ell}_t - \bar{\ell}_t)$ and set $\tau = \alpha\beta_1\gamma$ for some parameter $\gamma > 0$. Then, for $\eta = 1$, Eq. 13 is equivalent to the following,*

$$\begin{aligned}
\bar{s}_{t+1} &\leftarrow \beta_1 \bar{s}_t + (1-\beta_1)\hat{\ell}_t + \beta_1\gamma(\hat{\ell}_t - \hat{\ell}_{t-1}) \\
q_{t+1} &\leftarrow q_t^{1-\alpha} \exp\left(-\bar{s}_{t+1}\right)^{\alpha},
\end{aligned} \tag{14}$$

after eliminating the auxiliary q_{t+1} and relabeling q^{in} as q.

A proof is in Sec. B.3. We remark that it is also possible to obtain an update that generalizes SARAH (Nguyen et al., 2017) by setting $\beta_1 = \gamma = 1$, replacing the old site $\widehat{\ell}_{t-1}$ with a fresh one on the current data $\widehat{\ell}_{t|t-1}$ and periodically resetting the recursive estimator to the full gradient.

For the isotropic Gaussian family, the VRM update can recover MARS-momentum exactly.

Lemma 2. *Consider Eq. 14 for the isotropic Gaussian family of the form $q = \mathcal{N}(\boldsymbol{\theta} \,|\, \mathbf{m}, \mathbf{I})$. If we compute the site using $\widehat{\mathbf{g}}_t$ evaluated at the mean $\mathbf{m}_t$ of q_t, then Eq. 14 reduces to the MARS update (Yuan et al., 2025), that is, the STORM update of Eq. 3 where the last term is multiplied by γ and the old gradient $\widehat{\mathbf{g}}_{t-1}$ is reused rather than recomputed as $\widehat{\mathbf{g}}_{t|t-1}$, that is,*

$$\bar{\mathbf{g}}_{t+1} \leftarrow \beta_1 \bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t + \beta_1\gamma(\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t-1}). \tag{15}$$

Algorithm 1 Site-Momentum for IVON with SOM and VRM variants.

1: $\bar{\mathbf{g}} \leftarrow \beta_1 \bar{\mathbf{g}} + (1-\beta_1)\Big(\widehat{\mathbf{g}}_t + \gamma \frac{\beta_1}{1-\beta_1} \bar{\mathbf{h}}(\mathbf{m}_t - \mathbf{m}_{t-1}) \quad (\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t-1})\Big)$

2: $\bar{\mathbf{h}} \leftarrow \beta_3 \bar{\mathbf{h}} + (1-\beta_3)\ \widehat{\mathbf{g}}_t(\boldsymbol{\theta} - \mathbf{m}_t)/\boldsymbol{\sigma}^2$

3: $\mathbf{h} \leftarrow \beta_2 \mathbf{h} + (1-\beta_2)\bar{\mathbf{h}} \ + \frac{1}{2}(1-\beta_2)^2(\mathbf{h} - \bar{\mathbf{h}})^2/(\mathbf{h}+\delta)$

4: $\mathbf{m}_{t+1} \leftarrow \mathbf{m}_t - \alpha\, \mathrm{clip}\,[\ (\ \bar{\mathbf{g}}\ /\ (1-\beta_1^t) + \delta \mathbf{m}_t)/(\mathbf{h}+\delta)\ ;\ -\xi, \xi\ ]$

A proof is in Sec. B.4. We note that γ in MARS was introduced heuristically, as was reusing the old gradient $\widehat{\mathbf{g}}_{t-1}$ rather than $\widehat{\mathbf{g}}_{t|t-1}$ to speed up the algorithm (called MARS-approx.). Here, these arise from our theoretical derivation, where $\gamma = \tau/(\alpha\beta_1)$ is the posterior-correction strength. For the full Gaussian family, our update gives a MARS-like update with second-order momentum.

Lemma 3. *Consider Eq. 14 for full Gaussians $\mathcal{N}(\boldsymbol{\theta} \,|\, \mathbf{m}, \boldsymbol{\Sigma})$. Then, the momentum updates are:*

$$\bar{\mathbf{H}}_{t+1} = \beta_1 \bar{\mathbf{H}}_t + (1-\beta_1)\widehat{\mathbf{H}}_t + \beta_1 \gamma \left(\widehat{\mathbf{H}}_t - \widehat{\mathbf{H}}_{t-1}\right), \tag{16}$$

$$\bar{\mathbf{g}}_{t+1} = \beta_1 \bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t + \beta_1 \gamma \left(\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t-1}\right) + \beta_1(\bar{\mathbf{H}}_t - \gamma \widehat{\mathbf{H}}_{t-1})(\mathbf{m}_t - \mathbf{m}_{t-1}). \tag{17}$$

A proof is in Sec. B.5. For $\gamma = 0$, and setting the Hessian momentum to zero, this recovers a momentum similar to SGDHess (Tran & Cutkosky, 2022) (SGDHess uses the current Hessian). In that sense, this is a generalization to MARS-like second-order momentum.

Interestingly, the updates in Eq. 13 can also be understood from the viewpoint of a primal-dual optimization algorithm called the Bayesian Alternating Direction Method of Multipliers (Bayesian-ADMM) recently proposed by Möllenhoff et al. (2026). The site-functions play the role of Lagrange multipliers (dual variables) to add constraints on the distributions. Such connections between MARS and ADMM (Boyd et al., 2011) are so far unknown, and we describe these in Sec. C.

3.3 Practical Variants for Variational Learning at Scale

To run our new momentum at scale, we add our proposed second-order momentum (SOM) and variance-reduced momentum (VRM) to IVON (Shen et al., 2024, Algorithm 1) and EVON (Minut et al., 2026, Algorithm 2). We provide a detailed derivation and pseudocodes in Sec. D. Alg. 1 shows the required modifications to IVON (Shen et al., 2024, Algorithm 1) for adding either SOM or VRM. For EVON we essentially run the same IVON site-momentum algorithms but in the Eigenspace of Shampoo's preconditioner which requires projections that are detailed in Alg. 3.

The main issue with the method presented in Lemma 3 is the increase in memory requirements due to requiring the additional storage of $\widehat{\mathbf{H}}_{t-1}$ and $\mathbf{m}_{t-1}$ for the updates at time step t, in addition to the storage of the previous gradient estimate $\widehat{\mathbf{g}}_{t-1}$ that is used for gradient correction. In SOM, we therefore drop the gradient correction term and the additional storage of $\widehat{\mathbf{H}}_{t-1}$ by setting $\gamma = 0$ which leads to an SGDHess-like version that also corresponds to the formulation from Thm. 2. In SOM, we still downweight the additional Hessian term with a factor γ as we found it to work better. We also use a separately-tunable β_3 for the Hessian momentum term. For VRM, we set $\beta_3 = 0$ and only retain the gradient correction part from Lemma 3 to save memory. This leads to a MARS-like update for IVON and EVON. As we will show next, this version also performs better with larger γ than the $\gamma = 0.025$ that is recommended by Yuan et al. (2025).

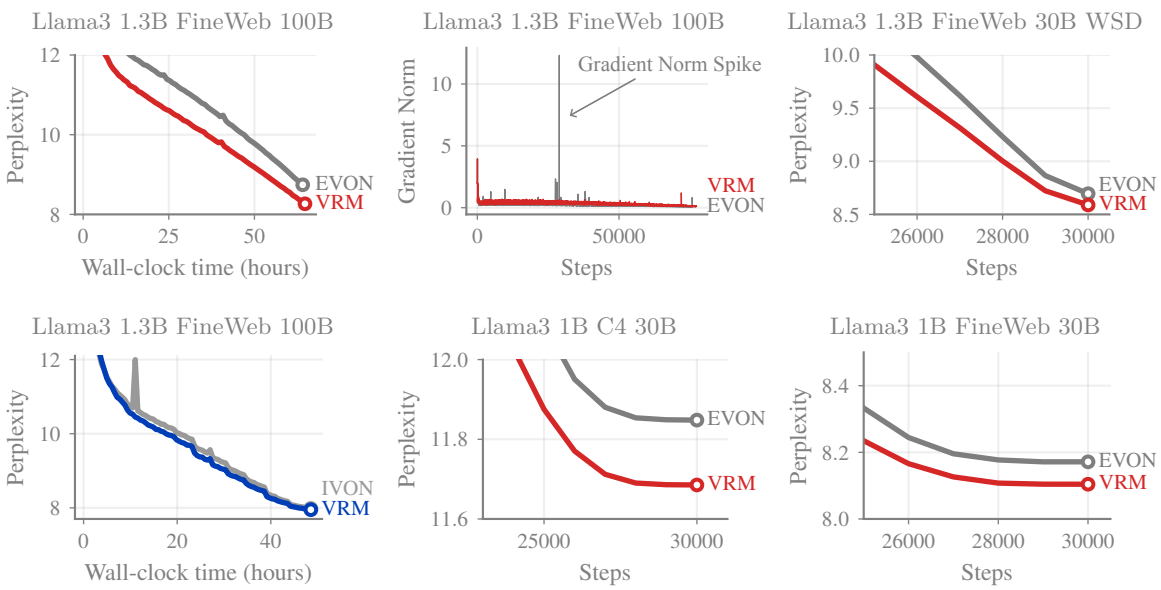


Figure 2: Llama3 experiments with batch size 1M. We tuned hyper-parameters for EVON, and then added VRM. VRM does not produce loss spike or gradient norm spikes. VRM improves perplexity over baseline EVON, for example, by 0.76 points at step 30k for Llama3 1.3B on FineWeb 100B. Improvement is consistent when using other variational optimizers or data settings.

4 Experiments

We focus our experiments on language model pre-training using the Llama (Touvron et al., 2023) architecture by reusing the publicly available code from Minut et al. (2026). We primarily train Llama models of 936M (1B) parameters and 1,339M (1.3B) parameters. We use C4 (Raffel et al., 2020) and FineWeb (Penedo et al., 2024) and train on up to 100B tokens on as many as 64 GPUs in parallel with a batch size of about 1M tokens. To the best of our knowledge, this is the first result showing a successful and stable variational learning at this scale. We adapt the IVON and EVON optimizers based on the code from Shen et al. (2024) and Minut et al. (2026) to add SOM and VRM. We also adopt their tuned hyper-parameters for IVON and EVON, see Sec. E for details.

For VRM, we mainly use $\gamma = 0.25$ which differs from the $\gamma = 0.025$ recommended in MARS (Yuan et al., 2025). In our initial runs at smaller scale size, we did not find such larger values of γ to be as beneficial, for example, when training a 30M parameter GPT model on 3k steps with token batch size of about 60K. There, $\gamma = 0.025$ appeared better. However, this trend drastically changed when moving to larger models, batch sizes, and longer training durations, where $\gamma = 0.25$ was significantly better. In our experience, drawing conclusion too early may be the reason why these methods do not appear to work well at first. It is advisable therefore to thoroughly explore the larger settings.

For SOM we generally used larger scales around $\gamma = 1.0$. However, we found that SOM presented only marginal improvements; see Fig. 3 for such a result. Therefore, we focus our following discussion on VRM which we found to improve both the convergence and final performance, and also stabilize training. More results and ablations are in App. F.

VRM improves performance at no runtime cost. Our main results are summarized in Fig. 2 on 1.3B and 1B models. Throughout different scales of training data from 30B to 100B tokens, different datasets and learning rate schedules, namely cosine decay (Loshchilov & Hutter, 2017) and warmup-stable-decay (WSD) (Zhai et al., 2022; Hu et al., 2024), we find that adding VRM improves both the convergence and final performance as measured in terms of validation perplexity.

For example, in the top left panel we train a 1.3B model on 100B tokens from FineWeb using cosine decay, though this run did not finish training by the submission deadline. The performance is improved by up to around 0.8 points in perplexity during training but, notably, at no additional cost in terms of runtime. This is different from recent variance reduction works for variational learning (Daheim et al., 2026) which only obtain such improvements under significant runtime overhead. The improvements also transfer to, for example, the end of training for a WSD setting on the top right with about 0.15 points improvement or the two 1B runs on C4 and FineWeb on the bottom with improvements of about 0.2 points. Notably, these improvements appear stronger on the recently-proposed EVON optimizer than on IVON, which is shown on the bottom left. This is interesting as we observed EVON to be more stable than IVON in general.

VRM stabilizes training. Interestingly, improvements also show in the gradient norms, where we empirically observe that VRM seems to have fewer gradient norm spikes than the baselines, especially for EVON. For EVON, we show this in the center-most panel on the top, where EVON, which is plotted in the background, exhibits multiple strong spikes that are mostly absent in VRM. In these experiments, we fixed the order of the training data such that these effects would show in both methods if they were to be purely attributed to data. For IVON, similar spikes are observed in the bottom left, where there is also a loss spike at roughly 15 hours of training.

VRM can be started after warmup but decay makes a difference. Daheim et al. (2026) have shown that variance reduction can be started later in training to reduce cost but with similar performance. Although we do not increase runtime cost, we still experiment with this approach, as we have found that the gradient correction term is particularly unstable in very early stages of training that roughly correspond to the warmup phase of a WSD schedule, for example. This is plotted in Fig. 4 for a training run with EVON and a 1.3B model. We find in Fig. 7 that starting correction later after 2k steps can be better than starting at step 0.

However, the gains in final performance reduce during the decay phase, which is particularly visible in the WSD plot in the top right. There, improvements shrink from roughly 0.4 during the stable phase of training to roughly 0.1 at the end. The decrease in gap starts at the same time as the decay. This is also true for a cosine decay schedule, where the gap closes at later stages of training, although final performance is, in our experiments, still always better for VRM.

5 Conclusion

We have derived a new site-momentum for natural-gradient variational learning. This fills an important gap in variational learning which largely relied on Adam-like momentum more suitable for non-Bayesian settings. We have connected the new site-momentum to various techniques from the optimization literature like SGDHess, and also to momentum-based variance reduction techniques like STORM and MARS. We have also derived a practical site-momentum that works well on large scale deep learning problems like pretraining a 1.3B LLM on up to 100B tokens with a 1M token batch size, where the new method improves both convergence and final performance.

AI Use Statement

No AI was used in the process of writing this paper and we did not use any AI for ideating or for implementing our experiments.

Reproducibility Statement

Our implementation is based only on publicy-available resources that can be found in Shen et al. (2024) and Minut et al. (2026). We adapt their code to ensure reproducibility by fixing the seeding of all data-loaders and of the random noise samplers that are used to sample from the posterior during training. This way, each run is ensured to have the exact same data order, even across different methods. Re-started runs will result in the same noise sampling. We list all hyper-parameters of our runs in Tab. 1 and throughout our discussion of the results and experimental set-up in Sec. E. In addition, we will release the full code to reproduce our experiments as open-source.

References

Dimitri P Bertsekas. Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. *Optimization for Machine Learning*, 2010(1-38):3, 2011. 2

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *International Conference on Machine Learning (ICML)*, 2015. 1

Stephen P. Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, 2011. 7

Bai Cong, Nico Daheim, Yuesong Shen, Rio Yokota, Mohammad Emtiyaz Khan, and Thomas Möllenhoff. Improving LoRA with variational learning. *arXiv preprint arXiv:2506.14280*, 2025. 1

Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex SGD. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1, 3, 5

Nico Daheim, Thomas Möllenhoff, Ming Liang Ang, and Mohammad Emtiyaz Khan. SVRG and beyond via posterior correction. In *International Conference on Machine Learning (ICML)*, 2026. 1, 4, 6, 9, 20

Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011. 1

Shengding Hu, Yuge Tu, Xu Han, Ganqu Cui, Chaoqun He, Weilin Zhao, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Xinrong Zhang, Zhen Leng Thai, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, dahai li, Zhiyuan Liu, and Maosong Sun. MiniCPM: Unveiling the potential of small language models with scalable training strategies. In *First Conference on Language Modeling*, 2024. 9

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013. 3

Mohammad Emtiyaz Khan. Knowledge Adaptation as Posterior Correction. *arXiv preprint arXiv:2506.14262*, 2025. 1, 4, 6, 13

Mohammad Emtiyaz Khan and Haavard Rue. The Bayesian learning rule. *J. Mach. Learn. Res. (JMLR)*, 24(281):1–46, 2023. 1, 3, 13, 17

Mohammad Emtiyaz Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable Bayesian deep learning by weight-perturbation in Adam. In *International Conference on Machine Learning (ICML)*, 2018. 4

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 3

Eren Mehmet Kiral, Thomas Möllenhoff, and Mohammad Emtiyaz Khan. The Lie-Group Bayesian learning rule. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2023. 4

Wu Lin, Mark Schmidt, and Mohammad Emtiyaz Khan. Handling the positive-definite constraint in the Bayesian learning rule. In *International Conference on Machine Learning (ICML)*, 2020. 4, 17

Wu Lin, Valentin Duruisseaux, Melvin Leok, Frank Nielsen, Mohammad Emtiyaz Khan, and Mark Schmidt. Simplifying momentum-based positive-definite submanifold optimization with applications to deep learning. In *International Conference on Machine Learning (ICML)*, 2023. 4

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. 9

Adrian Robert Minut, Nico Daheim, Marco Miani, Mohammad Emtiyaz Khan, Wu Lin, and Thomas Möllenhoff. SOAP-Bubbles: Structured weight uncertainty for neural networks. *arXiv:2606.23357*, 2026. 1, 2, 4, 7, 8, 10, 17, 18, 21

Thomas Möllenhoff, Siddharth Swaroop, Finale Doshi-Velez, and Mohammad Emtiyaz Khan. Federated ADMM from Bayesian duality. In *International Conference on Learning Representations (ICLR)*, 2026. 4, 7, 15

Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: a novel method for machine learning problems using stochastic recursive gradient. In *International Conference on Machine Learning (ICML)*, 2017. 6

Keigo Nishida, Eren Mehmet KIRAL, Kenichi Bannai, Mohammad Emtiyaz Khan, and Thomas Möllenhoff. Log-normal multiplicative dynamics for stable low-precision deep learning. In *International Conference on Machine Learning (ICML)*, 2026. 1

Kazuki Osawa, Siddharth Swaroop, Mohammad Emtiyaz Khan, Anirudh Jain, Runa Eschenhagen, Richard E Turner, and Rio Yokota. Practical deep learning with bayesian principles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1, 4

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. 8

Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. 1

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. 8

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. 1

Andrei Semenov, Matteo Pagliardini, and Martin Jaggi. Benchmarking optimizers for large language model pretraining. *arXiv:2509.01440*, 2025. 3

Yuesong Shen, Nico Daheim, Bai Cong, Peter Nickl, Gian Maria Marconi, Clement Bazan, Rio Yokota, Iryna Gurevych, Daniel Cremers, Mohammad Emtiyaz Khan, and Thomas Möllenhoff. Variational learning is effective for large deep networks. In *International Conference on Machine Learning (ICML)*, 2024. 1, 2, 4, 7, 8, 10, 16, 17

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning (ICML)*, 2013. 1, 2

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. 2, 8

Hoang Tran and Ashok Cutkosky. Better SGD using second-order momentum. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1, 3, 4, 5, 7, 14, 16

Yida Yin, Zhiqiu Xu, Zhiyuan Li, Trevor Darrell, and Zhuang Liu. A coefficient makes SVRG effective. In *International Conference on Learning Representations (ICLR)*, 2025. 3

Huizhuo Yuan, Yifeng Liu, Shuang Wu, Zhou Xun, and Quanquan Gu. MARS: Unleashing the Power of Variance Reduction for Training Large Models. In *International Conference on Machine Learning (ICML)*, 2025. 1, 3, 6, 7, 8, 21

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 9

A Issues with the Existing Momentum for BLR

Here, we rewrite Eq. 6 to show that it includes the entropy term in the momentum. To arrive there, we first write the update Eq. 6 in terms of the displacement $\mathbf{d}_{t+1} = \boldsymbol{\lambda}_{t+1} - \boldsymbol{\lambda}_t$,

$$\mathbf{d}_{t+1} = -\rho \left(\boldsymbol{\lambda}_t + \tilde{\mathbf{g}}_t - \frac{\beta}{\rho} \mathbf{d}_t \right). \tag{18}$$

We introduce a new variable $\tilde{\boldsymbol{\xi}}_{t+1}$ for the term inside the brackets:

$$\tilde{\boldsymbol{\xi}}_{t+1} = \boldsymbol{\lambda}_t + \tilde{\mathbf{g}}_t - \frac{\beta}{\rho} \mathbf{d}_t, \qquad \mathbf{d}_{t+1} = -\rho \tilde{\boldsymbol{\xi}}_{t+1}. \tag{19}$$

The first equation can be written purely in terms of $\tilde{\boldsymbol{\xi}}$,

$$\tilde{\boldsymbol{\xi}}_{t+1} = \boldsymbol{\lambda}_t + \tilde{\mathbf{g}}_t + \beta \tilde{\boldsymbol{\xi}}_t. \tag{20}$$

Defining $\boldsymbol{\xi} = (1-\beta)\tilde{\boldsymbol{\xi}}$ and multiplying the first equation with $(1-\beta)$ we obtain,

$$\boldsymbol{\xi}_{t+1} = \beta\boldsymbol{\xi}_t + (1-\beta)(\boldsymbol{\lambda}_t + \tilde{\mathbf{g}}_t), \qquad \boldsymbol{\lambda}_{t+1} = \boldsymbol{\lambda}_t - \alpha\boldsymbol{\xi}_{t+1}, \tag{21}$$

where $\alpha = \rho/(1-\beta)$. We see that compared to Eq. 9, this includes the contribution $\boldsymbol{\lambda}_t$ from the entropy in the momentum.

B Derivation for Site-Momentum

B.1 Proof of Thm. 1

For the isotropic Gaussian family $q(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \,|\, \mathbf{m}, \mathbf{I})$, we have $\mathbf{T}(\boldsymbol{\theta}) = \boldsymbol{\theta}$, $\boldsymbol{\lambda} = \boldsymbol{\mu} = \mathbf{m}$ and base measure $h(\boldsymbol{\theta}) = \exp(-\frac{1}{2}\boldsymbol{\theta}^\top\boldsymbol{\theta})$. Moreover, the stochastic natural gradient is $\widehat{\mathbf{g}}_t := \nabla\ell_t(\boldsymbol{\theta})$, where $\ell_t(\boldsymbol{\theta})$ is the minibatch loss at step t that also includes a contribution from the prior and $\boldsymbol{\theta} \sim q_t$. The form in Eq. 9 assumes a constant base measure. Since the base measure is non-constant, we use the more general form in Khan & Rue (2023, Eq. 6) which for our momentum-setting reads:

$$\boldsymbol{\lambda}_{t+1} \leftarrow (1-\alpha)\boldsymbol{\lambda}_t - \alpha\left[\bar{\boldsymbol{\lambda}}_{t+1} + \tilde{\nabla}\mathbb{E}_{q_t}[\log h(\boldsymbol{\theta})]\right]. \tag{22}$$

Substituting the above forms into Eq. 9 and Eq. 22, we get the following updates:

$$\bar{\mathbf{g}}_{t+1} \leftarrow \beta_1\bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t, \tag{23}$$

$$\mathbf{m}_{t+1} \leftarrow \mathbf{m}_t - \alpha\bar{\mathbf{g}}_{t+1}. \tag{24}$$

This fits the form of the classical momentum in Eq. 2 and exactly recovers it if, instead of sampling $\boldsymbol{\theta} \sim q_t$, we use an approximation $\boldsymbol{\theta} = \mathbf{m}_t$.

B.2 Proof of Thm. 2 and lemma 1

To obtain the results in Thm. 2, we write out Eq. 9 for a general Gaussian family $q(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \,|\, \mathbf{m}, \boldsymbol{\Sigma})$. We use the formulation in the right-hand side of Eq. 9 that uses site-functions. For Gaussians, by Khan (2025, Table 1), the loss site-functions have the following form:

$$\widehat{\ell}_t(\boldsymbol{\theta}) = \widehat{\mathbf{g}}_t^\top\boldsymbol{\theta} + \tfrac{1}{2}(\boldsymbol{\theta}-\mathbf{m})^\top\widehat{\mathbf{H}}_t(\boldsymbol{\theta}-\mathbf{m}), \quad \widehat{\mathbf{g}}_t = \nabla\ell_t(\boldsymbol{\theta}'), \quad \widehat{\mathbf{H}}_t = \nabla^2\ell_t(\boldsymbol{\theta}'), \tag{25}$$

where ℓ_t is the minibatch loss at step t that also includes a contribution from the prior and $\boldsymbol{\theta}' \sim q_t$. We use the same form for the momentum site-function $\bar{\ell}(\boldsymbol{\theta})$ and parametrize it by $\bar{\mathbf{g}}$ and $\bar{\mathbf{H}}$. In order to conveniently add old and new site-functions in Eq. 9, we rewrite the previous $\bar{\ell}_t(\boldsymbol{\theta})$ so that the quadratic term is centered at $\mathbf{m}_t$ rather than at $\mathbf{m}_{t-1}$:

$$\bar{\ell}_t(\boldsymbol{\theta}) = \bar{\mathbf{g}}_t^\top\boldsymbol{\theta} + \tfrac{1}{2}(\boldsymbol{\theta}-\mathbf{m}_{t-1})^\top\bar{\mathbf{H}}_t(\boldsymbol{\theta}-\mathbf{m}_{t-1}) \tag{26}$$

$$= \bar{\mathbf{g}}_t^\top\boldsymbol{\theta} + \tfrac{1}{2}(\boldsymbol{\theta}-\mathbf{m}_t)^\top\bar{\mathbf{H}}_t(\boldsymbol{\theta}-\mathbf{m}_t) + \boldsymbol{\theta}^\top\bar{\mathbf{H}}_t(\mathbf{m}_t-\mathbf{m}_{t-1}) + \text{const.} \tag{27}$$

The constant term does not depend on $\boldsymbol{\theta}$ and does not affect the update-step as the constant is absorbed by the normalization.

Using the above reformulation in Eq. 27, addition and scalar multiplication of the site functions in Eq. 9 becomes addition and scalar multiplication on the $\mathbf{g}$, $\mathbf{H}$ variables, which directly yields:

$$\bar{\mathbf{g}}_{t+1} = \beta_1(\bar{\mathbf{g}}_t + \bar{\mathbf{H}}_t(\mathbf{m}_t-\mathbf{m}_{t-1})) + (1-\beta_1)\widehat{\mathbf{g}}_t \tag{28}$$

$$= \beta_1\bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t + \beta_1\bar{\mathbf{H}}_t(\mathbf{m}_t-\mathbf{m}_{t-1}) \tag{29}$$

$$\bar{\mathbf{H}}_{t+1} = \beta_1\bar{\mathbf{H}}_t + (1-\beta_1)\widehat{\mathbf{H}}_t. \tag{30}$$

This concludes the proof of Thm. 2. Giving Eq. 30 its own momentum coefficient and setting that to zero, yields $\bar{\mathbf{H}}_t = \widehat{\mathbf{H}}_{t-1}$. Approximating the expectations with a sample at the mean, that is, $\boldsymbol{\theta}' = \mathbf{m}$, we get the following:

$$\bar{\mathbf{g}}_{t+1} = \beta_1 \bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\nabla}\ell_t(\mathbf{m}_t) + \beta_1 \nabla^2 \ell_{t-1}(\mathbf{m}_{t-1})(\mathbf{m}_t - \mathbf{m}_{t-1}). \tag{31}$$

Replacing $\nabla^2 \ell_{t-1}(\mathbf{m}_{t-1})$ with $\nabla^2 \ell_t(\mathbf{m}_t)$ gives the momentum of Tran & Cutkosky (2022).

B.3 Proof of Thm. 3

Setting $\eta = 1$, substituting the form of q_{t+1} into the update for q^{in}_{t+1} and relabeling q^{in} as q yields,

$$q_{t+1} \leftarrow q_t^{1-\alpha} \exp\left(-\alpha(\bar{\ell}_{t+1} + \beta_1 \gamma(\widehat{\ell}_t - \bar{\ell}_t))\right) \tag{32}$$

$$= q_t^{1-\alpha} \exp\left(-\bar{s}_{t+1}\right)^{\alpha}, \tag{33}$$

where the second step follows from the definition of $\bar{s}_{t+1}$. Then, we calculate

$$\bar{s}_{t+1} - \beta_1 \bar{s}_t = \bar{\ell}_{t+1} - \beta_1 \bar{\ell}_t + \beta_1 \gamma \left[\widehat{\ell}_t - \bar{\ell}_t - \beta_1(\widehat{\ell}_{t-1} - \bar{\ell}_{t-1})\right] \tag{34}$$

$$= (1-\beta_1)\widehat{\ell}_t + \beta_1 \gamma \left[\widehat{\ell}_t - \beta_1 \bar{\ell}_{t-1} - (1-\beta_1)\widehat{\ell}_{t-1} - \beta_1(\widehat{\ell}_{t-1} - \bar{\ell}_{t-1})\right] \tag{35}$$

$$= (1-\beta_1)\widehat{\ell}_t + \beta_1 \gamma \left[\widehat{\ell}_t - \widehat{\ell}_{t-1}\right]. \tag{36}$$

Rearranging, this yields the update in $\bar{s}_{t+1}$ from the Thm. 3.

B.4 Proof of Lemma 2

The derivation follows as in Sec. B.1. Loss-sites are linear $\widehat{\ell}_t(\boldsymbol{\theta}) = \widehat{\mathbf{g}}_t^\top \boldsymbol{\theta}$, $\widehat{\mathbf{g}}_t = \nabla \ell_t(\boldsymbol{\theta}')$, $\boldsymbol{\theta}' \sim q_t$. These can be approximated as $\widehat{\mathbf{g}}_t = \nabla \ell_t(\mathbf{m})$. Arithmetic on the sites straightforwardly turns into arithmetic on $\bar{\mathbf{g}}$ and $\widehat{\mathbf{g}}$, which directly gives the claim of Lemma 2.

B.5 Proof of Lemma 3

The derivation proceeds the same as in Sec. B.2. Additionally to $\bar{s}_t$, we can center the old loss-site $\widehat{\ell}_{t-1}$ at the new point $\mathbf{m}_t$. For $\widehat{\ell}_{t-1}(\boldsymbol{\theta})$, this yields,

$$\widehat{\ell}_{t-1}(\boldsymbol{\theta}) = \widehat{\mathbf{g}}_{t-1}^\top \boldsymbol{\theta} + \tfrac{1}{2}(\boldsymbol{\theta} - \mathbf{m}_{t-1})^\top \widehat{\mathbf{H}}_{t-1}(\boldsymbol{\theta} - \mathbf{m}_{t-1}) \tag{37}$$

$$= \widehat{\mathbf{g}}_{t-1}^\top \boldsymbol{\theta} + \tfrac{1}{2}(\boldsymbol{\theta} - \mathbf{m}_t)^\top \widehat{\mathbf{H}}_{t-1}(\boldsymbol{\theta} - \mathbf{m}_t) + \boldsymbol{\theta}^\top \widehat{\mathbf{H}}_{t-1}(\mathbf{m}_t - \mathbf{m}_{t-1}) + \text{const.} \tag{38}$$

Arithmetic on the sites directly translates to the $\mathbf{g}$ and $\mathbf{H}$ parameters in Sec. B.2, which gives:

$$\bar{\mathbf{H}}_{t+1} = \beta_1 \bar{\mathbf{H}}_t + (1-\beta_1)\widehat{\mathbf{H}}_t + \gamma\beta_1 \left(\widehat{\mathbf{H}}_t - \widehat{\mathbf{H}}_{t-1}\right), \tag{39}$$

$$\bar{\mathbf{g}}_{t+1} = \beta_1(\bar{\mathbf{g}}_t + \bar{\mathbf{H}}_t(\mathbf{m}_t - \mathbf{m}_{t-1})) \tag{40}$$

$$+ (1-\beta_1)\widehat{\mathbf{g}}_t + \gamma\beta_1 \left(\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t-1} - \widehat{\mathbf{H}}_{t-1}(\mathbf{m}_t - \mathbf{m}_{t-1})\right). \tag{41}$$

Rearranging gives the result of Lemma 3.

C VRM as a Bayesian Primal-Dual Algorithm

We assume that the loss includes the prior contribution, so that the objective in Eq. 4 is $\mathbb{E}_q[\ell] - \mathcal{H}(q)$, where $\mathcal{H}$ denotes entropy. We then use a principle called variable splitting, where the optimization over q is split into a *client* distribution $q^c(\boldsymbol{\theta})$ which handles the loss and prior, and a *server* distribution $q^s(\boldsymbol{\theta})$ which handles the entropy. The terminology of client and server is chosen to fit the federated learning setting of Möllenhoff et al. (2026), where multiple clients are used. Here, to recover VRM, we only use a single client. Both q^c and q^s are auxiliary copies of the same posterior q, giving the equivalent constrained problem

$$\min_{q^c, q^s \in \mathcal{Q}} \quad \mathbb{E}_{q^c}[\ell] - \mathcal{H}(q^s) \qquad \text{subject to } \boldsymbol{\mu}_c = \boldsymbol{\mu}_s, \tag{42}$$

where $\boldsymbol{\mu}_c = \mathbb{E}_{q^c}[\mathbf{T}(\boldsymbol{\theta})]$ is the expectation parameter. We assume a regular minimal exponential family with constant base measure, so equality of expectation parameters $\boldsymbol{\mu}_c = \boldsymbol{\mu}_s$ implies $q^c = q^s$ and Eq. 42 recovers the original objective.

Introducing a multiplier $\mathbf{a}$ for the constraint gives the Lagrangian,

$$\mathcal{L}(q^c, q^s, \mathbf{a}) = \mathbb{E}_{q^c}[\ell] - \mathcal{H}(q^s) + \mathbf{a}^\top(\boldsymbol{\mu}_s - \boldsymbol{\mu}_c) \tag{43}$$

$$= \mathbb{E}_{q^c}[\ell - \bar{\ell}] - \mathcal{H}(q^s) + \mathbb{E}_{q^s}[\bar{\ell}], \qquad \bar{\ell}(\boldsymbol{\theta}) = \langle \mathbf{a}, \mathbf{T}(\boldsymbol{\theta}) \rangle. \tag{44}$$

The loss-site represents the Lagrange multiplier for the equality constraint. Thus, loss-sites can be viewed as dual variables to the primal posterior variables q^s and q^c.

We now apply a linearized variant of Bayesian-ADMM (Möllenhoff et al., 2026, Eqs. 18–20) where the expected loss is replaced with its (linearized) site. Denote the server as $q_t^s = q_t^{\text{in}}$, and let $\widehat{\ell}_t$ be the site-function evaluated at q_t^s. We use separate client and server KL weights $\rho_c, \rho_s > 0$ and damp the dual step. With these choices, (Möllenhoff et al., 2026, Eqs. 18–20) read as follows:

$$q_{t+1}^c \leftarrow \arg\min_q \; \mathbb{E}_q[\widehat{\ell}_t - \bar{\ell}_t] + \rho_c \mathbb{D}_{\text{KL}}(q \parallel q_t^s), \tag{45}$$

$$\bar{\ell}_{t+1} \leftarrow \bar{\ell}_t + (1-\beta_1)\rho_c\big(\log q_t^s - \log q_{t+1}^c\big), \tag{46}$$

$$q_{t+1}^s \leftarrow \arg\min_q \; -\mathcal{H}(q) + \mathbb{E}_q[\bar{\ell}_{t+1}] + \rho_s \mathbb{D}_{\text{KL}}\big(q \parallel q_{t+1}^c\big). \tag{47}$$

The dual damping in the second line is already used in practical Bayesian-ADMM, see (Möllenhoff et al., 2026, Alg. 1). The client problem has a closed-form solution, and inserting the solution into the dual update yields the site-EMA:

$$q_{t+1}^c \propto q_t^s \exp\left[-\frac{\widehat{\ell}_t - \bar{\ell}_t}{\rho_c}\right], \qquad \bar{\ell}_{t+1} \leftarrow \beta_1 \bar{\ell}_t + (1-\beta_1)\widehat{\ell}_t. \tag{48}$$

Setting $\alpha = (1+\rho_s)^{-1}$, the server solution (which is q_{t+1}^{in} in the main text) becomes

$$q_{t+1}^s \propto (q_{t+1}^c)^{\,1-\alpha} \exp[-\alpha \bar{\ell}_{t+1}] \propto (q_t^s)^{\,1-\alpha} \exp\Big[-\alpha\bar{\ell}_{t+1} - \tau(\widehat{\ell}_t - \bar{\ell}_t)\Big], \qquad \tau = \frac{1-\alpha}{\rho_c}. \tag{49}$$

This is exactly the VRM update for q_{t+1}^{in} in Eq. 13 for $\eta = 1$, when q_{t+1} is substituted. Through Thm. 3, it also yields the MARS-type site recurrence, establishing a connection between variance-reduced momentum (such as used in MARS) and Bayesian-ADMM (which can recover regular ADMM for isotropic Gaussians).

Finally, Thm. 3 gives the following correspondence between hyper-parameters:

$$\beta_1 \gamma = \frac{\tau}{\alpha} = \frac{\rho_s}{\rho_c}. \tag{50}$$

Thus, the VRM correction strength γ (which is also the tunable parameter introduced in MARS) is exactly proportional to the ratio of server and client KL weights providing a new interpretation for that parameter and motivating its adaptation during optimization as is often done in ADMM.

D Derivation of IVON & EVON with SOM and VRM

D.1 IVON with SOM and VRM (Alg. 2).

To derive the algorithm, we combine our new Gaussian site-momentum with the practical updates of IVON (Shen et al., 2024). Let ℓ denote the average data loss and fix the prior as $\pi_0(\boldsymbol{\theta}) = \mathcal{N}\big(\boldsymbol{\theta} \,\big|\, 0, (\kappa\delta)^{-1}\mathbf{I}\big)$. For IVON, we parametrize the posterior as (see line 4 in Alg. 2),

$$q_t(\boldsymbol{\theta}) = \mathcal{N}\big(\boldsymbol{\theta} \,\big|\, \mathbf{m}_t, \mathrm{diag}(\boldsymbol{\sigma}_t^2)\big)\,, \qquad \boldsymbol{\sigma}_t^2 = \frac{1}{\kappa(\mathbf{h}_t + \delta)}. \tag{51}$$

Here $\mathbf{h}_t$ is the accumulated loss-curvature and $\kappa(\mathbf{h}_t + \delta)$ is the posterior precision. In what follows, vector products, divisions, and powers are all elementwise.

At iteration t, draw $\boldsymbol{\theta}_t \sim q_t$ and compute (see lines 5 and 7 in Alg. 2)

$$\widehat{\mathbf{g}}_t = \frac{1}{B}\sum_{i\in\mathcal{B}_t} \nabla\ell_i(\boldsymbol{\theta}_t), \qquad \widehat{\mathbf{h}}_t = \widehat{\mathbf{g}}_t \odot \frac{\boldsymbol{\theta}_t - \mathbf{m}_t}{\boldsymbol{\sigma}_t^2}. \tag{52}$$

Stein's lemma (Gaussian integration by parts) gives,

$$\mathbb{E}\left[\widehat{\mathbf{h}}_t \mid q_t\right] = \mathbb{E}_{q_t}[\mathrm{diag}(\nabla^2\ell(\boldsymbol{\theta}))].$$

Thus IVON's estimator computes the diagonal curvature without an additional gradient evaluation or Hessian-vector product as needed in second-order momentum of Tran & Cutkosky (2022). Together, these two estimators give stochastic approximations of the Gaussian loss-site,

$$\widehat{d}_t(\boldsymbol{\theta}) = \widehat{\mathbf{g}}_t^\top\boldsymbol{\theta} + \frac{1}{2}(\boldsymbol{\theta} - \mathbf{m}_t)^\top \mathrm{diag}(\widehat{\mathbf{h}}_t)(\boldsymbol{\theta} - \mathbf{m}_t). \tag{53}$$

Since our sites also contain contribution from the prior, the overall site is $\widehat{\ell} = r + \widehat{d}_t$, where $r(\boldsymbol{\theta}) = \delta\|\boldsymbol{\theta}\|^2/2$. For appropriate initialization, the momentum in IVON's lines 6 and 7 can be written only in terms of $\widehat{d}_t$, and the prior is pushed into the update for the mean in line 9.

IVON-SOM. For the second-order momentum variant (b), we use Thm. 2 which directly specializes to the diagonal family. For practical SOM, we damp the Hessian term and give the Hessian-momentum an independent parameter β_3:

$$\begin{aligned}\bar{\mathbf{g}}_{t+1} &= \beta_1\bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t + \gamma\,\beta_1\bar{\mathbf{h}}_t \odot (\mathbf{m}_t - \mathbf{m}_{t-1}),\\ \bar{\mathbf{h}}_{t+1} &= \beta_3\bar{\mathbf{h}}_t + (1-\beta_3)\widehat{\mathbf{h}}_t.\end{aligned} \tag{54}$$

The choice $\gamma = 1$ and $\beta_3 = \beta_1$ recovers the diagonal site EMA from the paper exactly, but in practice, it can work better to set them independently. These together give lines 6(b) and 7(b) in the pseudocode.

IVON-VRM. To obtain VRM, we use a diagonal specialization of Lemma 3. Our practical VRM variant only uses the gradient-difference correction and omits both curvature-dependent corrections.

We also turn off the Hessian momentum. This avoids storing the previous mean, Hessian estimate and another variable for the Hessian momentum.

$$\bar{\mathbf{g}}_{t+1} = \beta_1 \bar{\mathbf{g}}_t + (1-\beta_1)\widehat{\mathbf{g}}_t + \gamma\,\beta_1(\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t-1}), \qquad \bar{\mathbf{h}}_{t+1} = \widehat{\mathbf{h}}_t. \tag{55}$$

These together give lines 6(a) and 7(a) in the pseudocode.

For either SOM or VRM, we can write the updated data site as

$$\bar{d}_{t+1}(\boldsymbol{\theta}) = \bar{\mathbf{g}}_{t+1}^\top \boldsymbol{\theta} + \frac{1}{2}(\boldsymbol{\theta} - \mathbf{m}_t)^\top \operatorname{diag}(\bar{\mathbf{h}}_{t+1})(\boldsymbol{\theta} - \mathbf{m}_t).$$

For such a site, the BLR step used in the main paper reads

$$q_{t+1} \propto q_t^{1-\rho} \exp\left(-\rho\kappa[r + \bar{d}_{t+1}]\right), \tag{56}$$

where we have moved the fixed prior/regularizer explicitly in the main update.

Let $\mathbf{s}_t = \boldsymbol{\sigma}_t^{-2} = \kappa(\mathbf{h}_t + \delta)$. Matching quadratic and linear coefficients gives

$$\mathbf{s}_{t+1} = (1-\rho)\mathbf{s}_t + \rho\kappa(\bar{\mathbf{h}}_{t+1} + \delta), \tag{57}$$

$$\mathbf{s}_{t+1} \odot \mathbf{m}_{t+1} = (1-\rho)\mathbf{s}_t \odot \mathbf{m}_t - \rho\kappa(\bar{\mathbf{g}}_{t+1} - \bar{\mathbf{h}}_{t+1} \odot \mathbf{m}_t). \tag{58}$$

Rearranging these, see for example Khan & Rue (2023, App. D), and substituting the precision parameterization yields,

$$\mathbf{h}_{t+1} = (1-\rho)\mathbf{h}_t + \rho\bar{\mathbf{h}}_{t+1}, \qquad \mathbf{m}_{t+1} = \mathbf{m}_t - \rho\,\frac{\bar{\mathbf{g}}_{t+1} + \delta\mathbf{m}_t}{\mathbf{h}_{t+1} + \delta}. \tag{59}$$

Adding practical modifications to these give us lines 8 and 9 in the pseudocode. The practical tricks are the same as in Shen et al. (2024). In particular, we replace the common step size by separate mean and precision learning rates, add the positive-precision correction (Lin et al., 2020), gradient debiasing and elementwise update clipping.

EVON with SOM and VRM (Alg. 3). EVON (Minut et al., 2026) runs IVON in a rotated space given by orthogonal matrices $\mathbf{Q}_L, \mathbf{Q}_R$. To obtain EVON-SOM and EVON-VRM, we can simply use the entire diagonal-Gaussian construction from the previous IVON derivation and follow the same steps as in (Minut et al., 2026, App. B) to run them in the rotated space. This yields the pseudocode in Alg. 3.

A caveat is that for SOM, the mean-displacement needs to be expressed in the rotated space,

$$\mathbf{Q}_L^\top(\mathbf{M}_t - \mathbf{M}_{t-1})\mathbf{Q}_R,$$

see line 5 (b).

VRM simply uses the gradient correction, but uses the projected gradients $\widehat{\mathbf{G}}_t^\circ - \widehat{\mathbf{G}}_{t-1}^\circ$ as we operate in the rotated space, see line 5(a).

Additionally to IVON's variables, EVON maintains second-moments to compute the rotated space.

$$\mathbf{L} \leftarrow \beta_S\mathbf{L} + (1-\beta_S)\widehat{\mathbf{G}}\widehat{\mathbf{G}}^\top, \qquad \mathbf{R} \leftarrow \beta_S\mathbf{R} + (1-\beta_S)\widehat{\mathbf{G}}^\top\widehat{\mathbf{G}},$$

and refreshes $\mathbf{Q}_L, \mathbf{Q}_R$ every K iterations. For an exact implementation, whenever the basis changes, one needs to remember the previous $\mathbf{Q}_L, \mathbf{Q}_R$ to express the cached previous gradient in the new coordinates before its next use.

Algorithm 2 Proposed Site-Momentum IVON with either VRM (a) or SOM (b)

Require: Learning rates $\{\alpha\}$; $\beta_1, \beta_2, \beta_3 \in [0, 1)$; $\delta, \kappa, h_0 > 0$; clip radius $\xi > 0$; damping $\gamma > 0$.

Initialize: $\mathbf{m}_0 \leftarrow \mathbf{m}_1 \leftarrow$ (NN weight init); $\mathbf{h} \leftarrow h_0$; $\bar{\mathbf{g}} \leftarrow 0$; (a) $\widehat{\mathbf{g}}_0 \leftarrow 0$; (b) $\bar{\mathbf{h}} \leftarrow 0$.
for $t = 1, 2, \ldots, m$ **do**
  Sample a mini-batch $\mathcal{B}$ of size B
  $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{m}_t, \boldsymbol{\sigma}^2)$ where $\boldsymbol{\sigma} \leftarrow 1/\sqrt{\kappa(\mathbf{h} + \delta)}$
  $\widehat{\mathbf{g}}_t \leftarrow \frac{1}{B} \sum_{i \in \mathcal{B}} \nabla \ell_i(\boldsymbol{\theta})$
  (a) $\bar{\mathbf{g}} \leftarrow \beta_1 \bar{\mathbf{g}} + (1 - \beta_1)(\widehat{\mathbf{g}}_t + \gamma \frac{\beta_1}{1-\beta_1}(\widehat{\mathbf{g}}_t - \widehat{\mathbf{g}}_{t-1}))$
  (b) $\bar{\mathbf{g}} \leftarrow \beta_1 \bar{\mathbf{g}} + (1 - \beta_1)(\widehat{\mathbf{g}}_t + \gamma \frac{\beta_1}{1-\beta_1} \bar{\mathbf{h}}(\mathbf{m}_t - \mathbf{m}_{t-1}))$
  (a) $\bar{\mathbf{h}} \leftarrow \widehat{\mathbf{g}}_t(\boldsymbol{\theta} - \mathbf{m}_t)/\boldsymbol{\sigma}^2$
  (b) $\bar{\mathbf{h}} \leftarrow \beta_3 \bar{\mathbf{h}} + (1 - \beta_3)\widehat{\mathbf{g}}_t(\boldsymbol{\theta} - \mathbf{m}_t)/\boldsymbol{\sigma}^2$
  $\mathbf{h} \leftarrow \beta_2 \mathbf{h} + (1 - \beta_2)\bar{\mathbf{h}} + \frac{1}{2}(1 - \beta_2)^2(\mathbf{h} - \bar{\mathbf{h}})^2/(\mathbf{h} + \delta)$
  $\mathbf{m}_{t+1} \leftarrow \mathbf{m}_t - \alpha \, \mathrm{clip}[\,(\,\bar{\mathbf{g}} \,/\, (1 - \beta_1^t) + \delta \mathbf{m}_t)/(\mathbf{h} + \delta)\,;\; -\xi, \xi\,]$
end for

Algorithm 3 Proposed Site-Momentum EVON with either VRM (a) or SOM (b).

Require: Learning rates $\{\alpha\}$; $\beta_1, \beta_2, \beta_3, \beta_S \in [0, 1)$; $\delta, \kappa, h_0 > 0$; damping $\gamma > 0$.

Initialize: $\mathbf{M}_0 \leftarrow \mathbf{M}_1 \leftarrow$ (NN weight init); $\mathbf{H} \leftarrow h_0$; $\bar{\mathbf{G}} \leftarrow 0$; (a) $\widehat{\mathbf{G}}_0^{\circ} \leftarrow 0$; (b) $\bar{\mathbf{H}} \leftarrow 0$.
for $t = 1, 2, \ldots, m$ **do**
  $\boldsymbol{\Theta} \leftarrow \mathbf{M}_t + \mathbf{Q}_L \mathbf{E} \mathbf{Q}_R^\top$ where $E_{i,j} \sim \mathcal{N}(0, \Sigma_{i,j})$, $\Sigma_{i,j} \leftarrow 1/(\kappa(H_{i,j} + \delta))$
  $\widehat{\mathbf{G}} \leftarrow \widehat{\nabla} \ell(\boldsymbol{\Theta})$, $\widehat{\mathbf{G}}_t^{\circ} \leftarrow \mathbf{Q}_L^\top \widehat{\mathbf{G}} \mathbf{Q}_R$
  (a) $\bar{\mathbf{G}} \leftarrow \beta_1 \bar{\mathbf{G}} + (1 - \beta_1)\left(\widehat{\mathbf{G}}_t^{\circ} + \gamma \frac{\beta_1}{1-\beta_1}(\widehat{\mathbf{G}}_t^{\circ} - \widehat{\mathbf{G}}_{t-1}^{\circ})\right)$
  (b) $\bar{\mathbf{G}} \leftarrow \beta_1 \bar{\mathbf{G}} + (1 - \beta_1)\left(\widehat{\mathbf{G}}_t^{\circ} + \gamma \frac{\beta_1}{1-\beta_1} \bar{\mathbf{H}} \odot (\mathbf{Q}_L^\top(\mathbf{M}_t - \mathbf{M}_{t-1})\mathbf{Q}_R)\right)$
  (a) $\bar{\mathbf{H}} \leftarrow \mathrm{clip}(\mathbf{E} \odot \widehat{\mathbf{G}}_t^{\circ}/\boldsymbol{\Sigma})$
  (b) $\bar{\mathbf{H}} \leftarrow \beta_3 \bar{\mathbf{H}} + (1 - \beta_3)\,\mathrm{clip}(\mathbf{E} \odot \widehat{\mathbf{G}}_t^{\circ}/\boldsymbol{\Sigma})$
  $\mathbf{H} \leftarrow \beta_2 \mathbf{H} + (1 - \beta_2)\bar{\mathbf{H}} + \frac{1}{2}(1 - \beta_2)^2(\mathbf{H} - \bar{\mathbf{H}})^{\odot 2}/(\mathbf{H} + \delta)$
  $\mathbf{U} \leftarrow (\bar{\mathbf{G}} + \delta \mathbf{Q}_L^\top \mathbf{M}_t \mathbf{Q}_R)/(\mathbf{H} + \delta)$
  $\mathbf{M}_{t+1} \leftarrow \mathbf{M}_t - \alpha \, \mathrm{clip}\big[\mathbf{Q}_L \mathbf{U} \mathbf{Q}_R^\top\big]$
  $\mathbf{L} \leftarrow \beta_S \mathbf{L} + (1 - \beta_S)\widehat{\mathbf{G}}\widehat{\mathbf{G}}^\top$
  $\mathbf{R} \leftarrow \beta_S \mathbf{R} + (1 - \beta_S)\widehat{\mathbf{G}}^\top \widehat{\mathbf{G}}$
  Every K steps:
    $\mathbf{Q}_L, \mathbf{Q}_R \leftarrow \mathrm{Eig}(\mathbf{L}), \mathrm{Eig}(\mathbf{R})$
    Update momentum $\bar{\mathbf{G}}$ (new basis)
end for

E Experimental Setup

For our Llama3 and NanoGPT training runs, we use the hyper-parameters found by Minut et al. (2026) as presented in Tab. 1. All training runs in the main manuscript with model sizes 1B and above use a batch size of 1984, and a sequence length of 512, which results in an effective batch size of about 1M. Unless otherwise noted, we use for VRM $\gamma = 0.25$, and for SOM $\gamma = 0.5$ and $\beta_3 = \beta_2$.

Table 1: Hyperparameters for NanoGPT and Llama3.

Optimizer	Model	κ	h_0	α	β_1	β_2	δ	clipping
IVON	NanoGPT 123M	4.66×10^{7}	1.80×10^{-2}	1.42×10^{2}	0.890	0.999877	1.72×10^{-5}	10^{-4}
	Llama3 135M	5.06×10^{8}	1.95×10^{-3}	1.18×10^{3}	0.826	0.999927	3.93×10^{-7}	2.49×10^{-4}
	Llama3 $\geq$ 135M	5.06×10^{8}	1.95×10^{-3}	6.00×10^{3}	0.826	0.999927	3.93×10^{-7}	2.49×10^{-4}
EVON	NanoGPT 123M	5.27×10^{6}	5.82×10^{-1}	1.72×10^{-2}	0.943	0.999088	1.00×10^{-6}	1.5
	Llama3 130M	7.61×10^{7}	9.31×10^{-1}	1.52×10^{-2}	0.881	0.999654	8.83×10^{-7}	1.5
	Llama3 $\geq$ 350M	7.61×10^{7}	1.00×10^{-5}	3.00×10^{-2}	0.881	0.999654	8.83×10^{-7}	1.5

Hardware and Multi-GPU Training All training runs were conducted on NVIDIA B200 nodes in parallel, connected locally through NVLINK and across nodes using Infiniband. Each node is equipped with 4 NVIDIA GPUs. Our training runs were ranging from using between 1, 8, and 16 nodes in parallel, which are 4, 32, and 64 Blackwell GPUs, respectively. We used only data parallelism, which gave us an efficiency of about 62% on 8 nodes compared to single node training. The 100B runs were run on 8 nodes / 32 GPUs in parallel, while most other runs were done using single node 4 GPUs. Two of our runs on 30B tokens were ran on 8 nodes / 64 GPUs, but these were not very compute efficient on only data parallelism, with an efficiency of 42% compared to single node / 4 GPU training.

F Ablation Studies and Additional Results

In this section, we present some additional ablations that are concerned with different training settings of VRM, as well as the behavior of the γ parameter.

IVON with SOM First, we test the performance of adding SOM to IVON on FineWeb 30B using a 1.3B Llama model trained for 30k steps with a batch size of 1M tokens. The perplexity is shown in Fig. 3. In this setting, we observe small improvements when using SOM that are below 0.05 points of perplexity when compared to the IVON baseline. Across other settings SOM did not match the performance gains of VRM and, therefore, we focus more on the latter in our experiments.

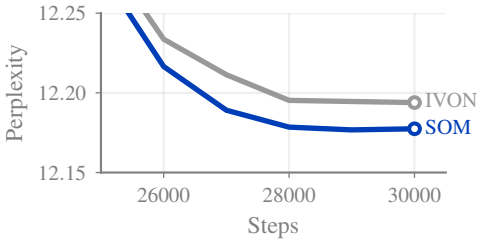


Figure 3: Llama3 1.3B trained on 30B using baseline IVON and IVON with SOM.

Exact Perplexity Improvements The exact improvements in terms of validation perplexity are shown in Tab. 2 for the Llama 1.3B models. Note, however, that these values may not be final, as some of the runs did not finish training at the submission deadline but will be updated once possible.

Table 2: Llama3 1.3B perplexity. Values can be earlier steps compared to plots in the main manuscript, as some runs were still running as of the submission deadline.

	EVON		IVON	
Setup	**Baseline**	**VRM**	**Baseline**	**VRM**
Fineweb 100B	9.322	9.126	10.273	9.658
Fineweb 30B	9.244	9.240	9.893	9.880
Fineweb 30B WSD	8.695	8.589	11.967	11.802
C4 30B	11.85	11.69	12.19	12.18

Observing the Gamma Term in VRM VRM introduces an additional gradient correction term scaled by γ into the momentum as follows: $\gamma \frac{\beta_1}{1-\beta_1}(\mathbf{g}_k - \mathbf{g}_{k-1})$. In Fig. 4, we plot the norm of this term (including the multiplication by γ) to observe how much it contributes to the optimizer step in comparison to the gradient norm. The gamma term is shown in Fig. 4, alongside the gradient norm of the same run. Here we observe that the gamma term rises sharply in the first 1000 steps, where the value even exceeds the gradient norm. Afterwards, the term stays stable and, especially when training with EVON, it does only very rarely exhibit spikes in some settings. This motivates us to downweigh the term early on by setting $\gamma = 0$. We think that, in the future, other schedules might be found that perform better, for example, increasing γ slowly similarly to how the weight of correction is increased in some of Daheim et al. (2026).

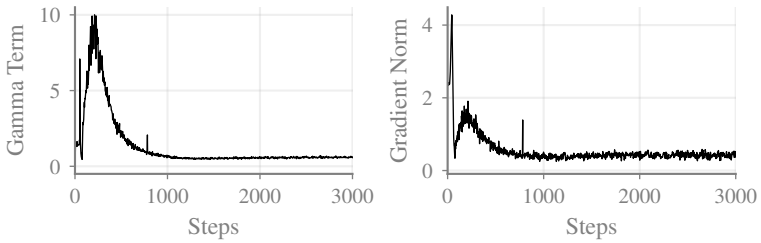


Figure 4: Left: Norm of the gamma term in EVON VRM on Llama3 1.3B trained on FineWeb 30B. Right: Gradient norm of the same run.

Gradient spikes and loss spikes in IVON We show a further result on a gradient spike observed in the EVON training of Llama3 1.B on FineWeb 100B in Fig. 5 (right). Another such result is found in Fig. 7 (right) , where VRM reeduces the number of gradient spikes compared to EVON when training a 1.3B model on 30B tokens of FineWeb with a WSD schedule.

EVON and VRM on Llama3 1B Here, we train a somewhat smaller version of Llama3 with 936M parameters on 100B tokens of FineWeb. The validation perplexity over the training trajectory of EVON and VRM training is shown in Fig. 5. The results are consistent with our main findings: VRM shows strong gains in perplexity over the baseline EVON.

Runs on smaller models below 0.5B In Fig. 6, we show results on smaller training settings, where the models have fewer parameters and training is done on fewer tokens. In particular, we train NanoGPT 123M on 1B Tokens, Llama3 134M, and Llama3 350M using IVON and EVON. In all cases, VRM outperforms the baseline IVON and EVON. In addition, we observe that $\gamma = 0.025$,

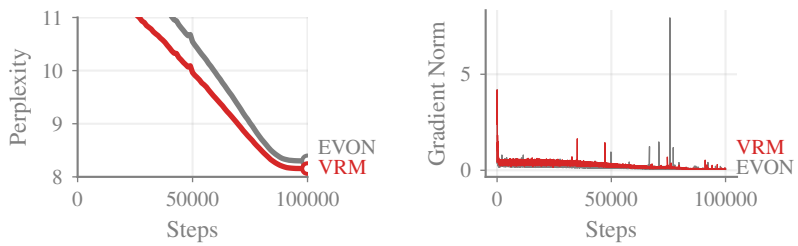


Figure 5: Llama3 1B on FineWeb, trained on 100B Tokens with a batch size of 1M. This run uses a slightly smaller model compared to the 1.3B model discussed in the main manuscript. (left) VRM again improves both convergence and final performance. (right) EVON has a large gradient spike that is absent in VRM.

as recommended by Yuan et al. (2025), performs well. All hyper-parameters were highly tuned by Minut et al. (2026), but despite that, adding VRM on top consistently results in a lower perplexity. Additionally, the NanoGPT run is a highly tuned speedrun[1], and yet VRM achieved a slight improvement.

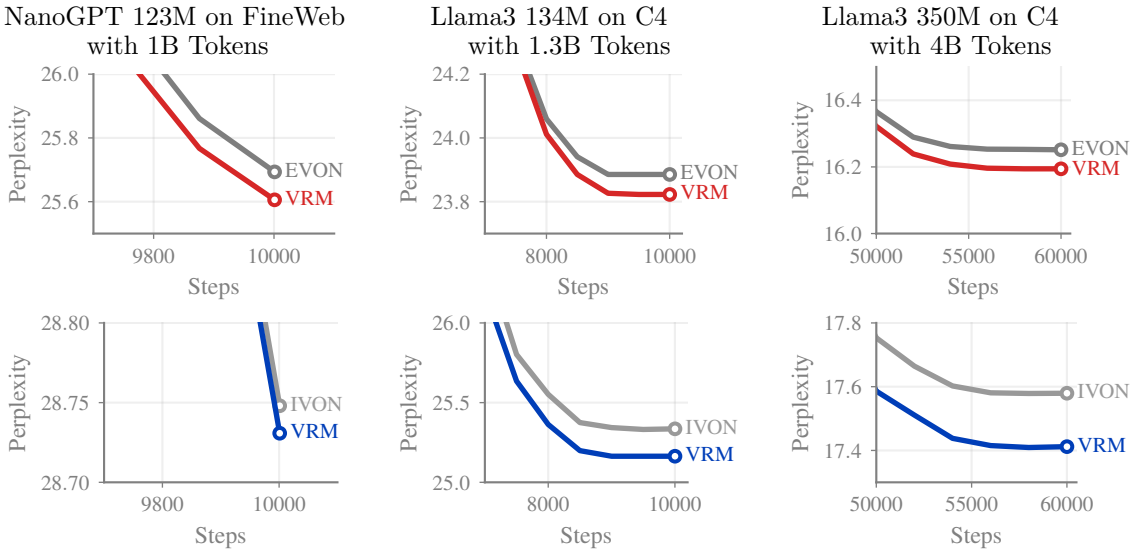


Figure 6: Runs on smaller versions of NanoGPT and Llama3. Across all runs, VRM consistently improves validation perplexity. We heavily tuned the hyper-parameters for the baseline optimizers IVON and EVON. VRM uses the same hyper-parameters, with the additional γ hyper-parameter set to 0.025, as originally proposed by Yuan et al. (2025) for MARS.

Starting VRM late Based on the observation that the gamma term in Fig. 4 can have spikes and rise sharply first 1k steps, we also try starting VRM late by setting $\gamma = 0$ for the first 2000 training steps. We compare this run to setting $\gamma = 0.25$ directly at the beginning of training of

[1]https://github.com/kellerjordan/modded-nanogpt

a Llama 1.3B model on FineWeb 30B using EVON with VRM and a batch size of 1M tokens as well as a warmup-stable-decay schedule. We show the results in Fig. 7 and find that $\gamma = 0.25$ right away provides poor initial evaluation perplexity and, while the run seems to catch up, also exhibits worse final evaluation perplexity.

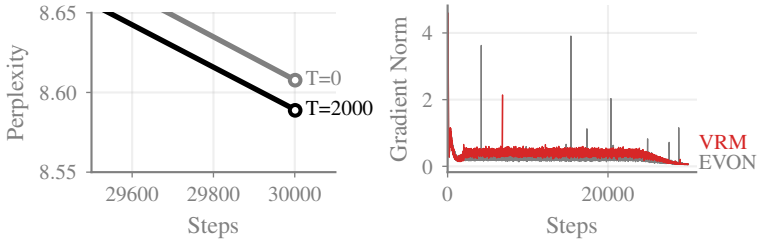


Figure 7: Left: Starting $\gamma = 0.25$ at step 0 performs worse than starting at step 2000 and setting $\gamma = 0$ before that on Llama3-1.3B with warmup stable decay on FineWeb 30B. Right: Gradient Norm shows spikes for baseline EVON, but less so for VRM.