3rd Symposium on Advances in Approximate Bayesian Inference, 2020 1–20

Gaussian Density Parametrization Flow: Particle and Stochastic Approaches

Théo Galy-Fajou GALY-FAJOU@TU-BERLIN.DE
TU Berlin
Valerio Perrone VPERRONE@AMAZON.COM
Amazon
Manfred Opper MANFRED.OPPER@TU-BERLIN.DE
TU Berlin

Abstract

Bayesian inference is intractable for most practical problems and requires approximation schemes with several trade-offs. Variational inference provides one of such approximations which, while powerful, has thus far seen limited use in high-dimensional applications due to its complexity and computational cost. This paper introduces a scalable, theoretically-grounded, and simple-to-implement algorithm for approximate inference with a variational Gaussian distribution. Specifically, we establish a practical particle-based algorithm to perform variational Gaussian inference that scales linearly in the problem dimensionality. We show that our approach performs on par with the state of the art on a set of challenging high-dimensional problems.

1. Introduction

Computing posterior distributions is intractable for many problems of interest. Markov chain Monte Carlo methods struggle to scale up to large datasets (Bardenet et al., 2017) and convergence is hard to diagnose (Cowles and Carlin, 1996). On the other hand, *Variational Inference* **(VI)** methods can rely on well-understood optimization techniques and scale well to large datasets, but the quality of the approximation depends heavily on the assumptions made. As most interesting real-world problems are typically high-dimensional, VI techniques either make crude approximations regarding the posterior distribution, such as mean-field (Hinton and van Camp, 1993) (every variable is assumed to be independent), or do not scale well with the number of dimensions. An increasingly popular class of solvers are particle-based variational methods (Gershman et al., 2012; Liu and Wang, 2016; Liu et al., 2016; Saeedi et al., 2017). Lying at the intersection of sampling and variational inference, these methods represent the variational distribution as an empirical distribution. They then optimize the particles positions instead of the distribution parameters.

In this work we introduce *Gaussian Particle Flow* **(GPF)**, a general-purpose algorithm to approximate a Gaussian variational distribution with particles. GPF leverages the *Gaussian Variational Approximation* **(GVA)** (Opper and Archambeau, 2009), a powerful tool to approximate intractable posterior distributions with multivariate Gaussians. *Gaussian Particle Flow* **(GPF)** is free of matrix inversions and second derivatives, scaling linearly with the dimensionality of the problem. We show that, under some conditions, the fixed-point solutions for Gaussians are exact and we have linear convergence. Finally, we provide empirical results both in fully-controlled synthetic settings and on a set of challenging real-world problems, demonstrating that our approach easily scales to high dimensions and achieves state-of-the-art performance.

2. Gaussian Particle Flow

2.1. Particle-Based Variational Inference

Bayesian inference aims to find the posterior distribution of a latent variable x given some observations y: $p(x|y) = \frac{p(y|x)p(x)}{p(y)}$. Even if the likelihood $p(y|x)$ and the prior $p(x)$ are known analytically, marginalizing out high-dimensional variables to compute $p(y)$ is typically intractable. VI approximates the posterior by the closest distribution within a tractable family, with closeness being measured by the *Kullback-Leibler* **(KL)** divergence. Denoting by $\mathcal{Q}$ a family of distributions, we look for $\arg\min_{q\in\mathcal{Q}} \mathrm{KL}\left[q(x)||p(x|y)\right]$. Since $p(y)$ is not known, we equivalently minimize the upper bound $\mathcal{F}$:

$$\mathrm{KL}\left[q(x)||p(x|y)\right] \leq \mathcal{F} = \mathbb{E}_q\left[p(y|x)p(x)\right] - \mathbb{H}_q, , \tag{1}$$

where $\mathbb{H}_q$ is the entropy of q. Note that $\mathcal{F}$ is known as the variational free energy and $-\mathcal{F}$ is known as the ELBO.

Stochastic gradient descent methods compute expectations (and gradients) with Monte Carlo samples from the current approximation of the variational density independently at each time step. Particle based methods for variational inference (ParVI) *draw samples once* at the beginning of the algorithm instead. They iteratively construct a transformation on an initial random variable (with a simple tractable density) leading to the minimization of the variational free energy. The iterative approach induces a deterministic temporal flow of random variables which depends on the current density of the variable itself. Using an approximation by the empirical density (which is represented by a the positions of a set of "particles") one obtains an interacting particle flow which converges asymptotically to an empirical approximation of the desired optimal variational density. A review of related work in VI and ParVI is given in Appendix A.

2.2. Gaussian density parametrization

We want to optimize the parameters θ of a Gaussian variational distribution $q_\theta(x)$ to closely approximate the target density $p(x) \propto \exp(-\varphi(x))$, where $x \in \mathbb{R}^D$. For example, for Bayesian inference, $\varphi(x) = -\log(p(y|x)) - \log(p(x))$. To this end, we minimize the free energy $\mathcal{F}$ described in Equation (1).

Gradient based minimization is facilitated by the so–called *reparametrization trick*. This replaces the expectation over the parameter dependent variational density q_θ by an expectation over a fixed density q_0 instead. In this way, unwanted derivatives of the type $\nabla_\theta q_\theta(x)$ are avoided. Any D dimensional random variable $x \sim q_\theta(x)$ can be constructed by a linear parametrization of a D-dimensional Gaussian random variable $x^0 \sim q^0 = \mathcal{N}(m^0, C^0)$:

$$x = \Gamma(x^0 - m^0) + m, \tag{2}$$

where $\Gamma \in \mathbb{R}^{D\times D}$ and $m \in \mathbb{R}^D$ are the variational parameters. The free energy is easily obtained as a function of the variational parameters:

$$\mathcal{F}(\Gamma, m) = -\log|\Gamma| + \mathbb{E}_{q_0}\left[\varphi(\Gamma(x^0 - m^0) + m)\right]. \tag{3}$$

To minimize $\mathcal{F}(\Gamma, m)$ with respect to its parameters, we define a gradient flow $\Gamma(t)$, $m(t)$ for $t \geq 0$ in parameter space. For the matrix valued parameter Γ, we apply a natural

gradient learning approach. This was introduced by S. Amari (1998, 7.6) as an efficient technique for learning the mixing matrix in models of blind source separation. We give a more detailed description on natural gradient theory in Appendix B. A combination of the natural gradient for Γ and an ordinary gradient for m leads to the following flow:

$$\frac{dm}{dt} = -\frac{\partial \mathcal{F}}{\partial m}, \quad \frac{d\Gamma}{dt} = -\frac{\partial \mathcal{F}}{\partial \Gamma}\Gamma^\top \Gamma, \tag{4}$$

where we have omitted the time dependency of the parameters. One can easily verify that the flow guarantees a decrease of $\mathcal{F}$ over time. The explicit form of the flows is given by

$$\frac{dm}{dt} = -\mathbb{E}_{q^0}\left[\nabla_x \varphi(x)\right], \quad \frac{d\Gamma}{dt} = -A\Gamma, \tag{5}$$

where $A = \left(\mathbb{E}_{q_0}\left[\nabla_x \varphi(x)(x-m)^\top\right] - I\right)$. Using Stein's lemma (Ingersoll, 1987), we can show that the fixed-point solution of Equations (5) equal the conditions for the optimal variational Gaussian distribution solution given in Opper and Archambeau (2009).

A standard algorithmic approach to deal with the flow equations would be to discretize Equations (5) in time, and estimate expectations by drawing independent samples from the fixed Gaussian q^0 at each time step. A stochastic gradient version of this algorithm would combine a small number of samples with a decreasing learning rate, we refer later to this approach as **GDP** and describe all algorithms steps in Algorithm 2 in Appendix D. Remarkably, this scheme is different from previous algorithms for Gaussian variational inference (e.g., see Opper and Archambeau (2009); Titsias and Lázaro-Gredilla (2014)) as no matrix inversions or second derivatives of the function φ are required. We will next introduce a particle based algorithm as an approximation of the flow instead.

2.3. Particle dynamics

The dynamics of the parameters m and Γ induces a corresponding flow for the random variable x via (2). Applying Equations (5) on Equation (2) we obtain:

$$\frac{dx}{dt} = \frac{d\Gamma}{dt}(x^0 - m^0) + \frac{dm}{dt} = -A(x-m) - \mathbb{E}_{q_0}\left[\nabla_x \varphi(x)\right]. \tag{6}$$

Note that at each time t, A, m and $\mathbb{E}_{q_0}\left[\nabla_x \varphi(x)\right]$ still depend on the parameters $\Gamma(t)$ and $m(t)$. In Appendix C, we also give an alternative approach solely based on linear dynamics to obtain Equation (6), without considering parameters like m and Γ. This dependency can be removed by *undoing the reparametrization* and re-introducing expectations over the induced density q_θ^t of $x(t)$ itself. The main idea of the particle approach is to approximate this Gaussian density by the empirical distribution $\hat{q}^t$ computed from N samples x_i^t, $i = 1, \ldots, N$. These are sampled initially from from the density q^0 at time $t = 0$ and are then propagated using the discretized dynamics (6):

$$x_i^{t+1} = x_i^t - \eta_1^t \mathbb{E}_{\hat{q}^t}\left[\nabla_x \varphi(x)\right] - \eta_2^t A^t (x_i^t - m^t), \tag{7}$$

where η_1^t and η_2^t are learning rates to avoid unstable steps and $m^t = \frac{1}{N}\sum_{i=1}^N x_i^t$. We give further comments on the use of different optimization schemes in Section 3. It is easy to see that the particle dynamics also minimizes the *empirical approximation of the free energy* (3) obtained by replacing the expectation over q^0 with its empirical counterpart $\hat{q}^0$.

2.4. Algorithm

The algorithm we propose is to sample N particles $\{x_1^0, \ldots, x_N^0\}$ where $x_i^0 \in \mathbb{R}^D$ from q^0 (which can be centered around the MAP), and iteratively optimize their positions using Equation (7). Once convergence is reached, i.e., $\frac{d\mathcal{F}}{dt} = 0$, we can easily work with the obtained empirical distribution $\hat{q}(x) = \frac{1}{N}\sum_{i=1}^N \delta_{x,x_i}$ or alternatively the Gaussian density it represents, namely $q(x) = \mathcal{N}(m, C)$, where $m = \frac{1}{N}\sum_{i=1}^N x_i$ and $C = \frac{1}{N}\sum_{i=1}^N (x_i - m)(x_i - m)^\top$. To draw samples from q, the empirical covariance C does not need to be computed, as we can obtain new samples by computing:

$$x = \frac{1}{\sqrt{N}}\sum_{i=1}^N (x_i - m)\xi_i + m, \tag{8}$$

where ξ_i are i.i.d. normal variables: $\xi_i \sim \mathcal{N}(0, 1)$.

All the inference steps are summarized in Algorithm 1 in Appendix D. Our algorithm does not require computing second order derivatives and, more crucially, it also does not involve matrix inversions (unlike GVA) and thus scales well with the dimensionality of the problem. The complexity of the method compares favorably to GVA. In GVA we would need to perform matrix inversions of complexity $\mathcal{O}(D^3)$, which we avoid. The computational bottleneck now lies in the gradients $g(x_i)$, adding a storing complexity of $\mathcal{O}(ND)$, and in the matrix multiplication of $A(x-m)$, which has a total computational complexity of $\mathcal{O}(N^2D)$. The direct consequence is that, for fewer particles than dimensions ($N \leq D+1$), we improve speed through a low-rank approximation of the covariance matrix.

Dynamics and fixed points for Gaussian targets We can obtain some exact theoretical results for the dynamics and the fixed points of our algorithm when the target is a multivariate Gaussian density. While such targets may seem like a trivial application, our analysis might still give some intuition about the performance for more complicated densities.

Theorem 1 *If the target density $p(x)$ is a D-dimensional multivariate Gaussian, only $D+1$ particles are needed for Algorithm 1 to converge to the exact target parameters.*

Proof: The proof is given in Appendix E.

Theorem 2 *For a target $p(x) = \mathcal{N}(x \mid \mu, \Lambda^{-1})$, where $x \in \mathbb{R}^D$, and $N \geq D+1$ particles, the continuous time limit of Algorithm 1 will converge linearly for both the mean and the trace of the precision matrix:*

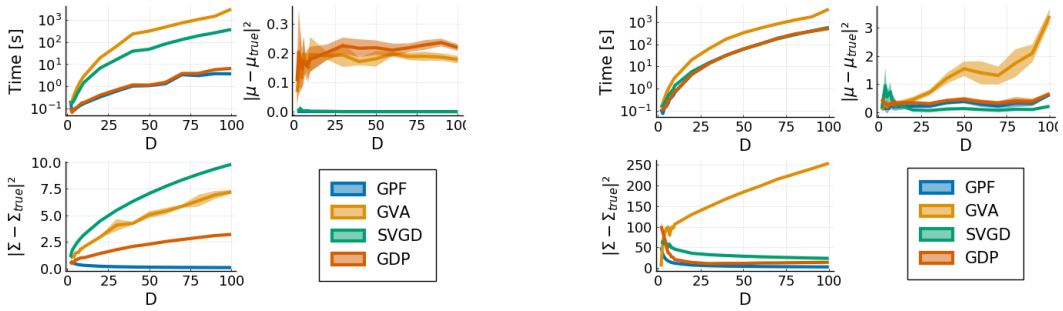
$$\begin{aligned} \|m^t - \mu\| &= e^{-\Lambda t}\|m^0 - \mu\|, \\ tr(\left(C^t\right)^{-1} - \Lambda) &= e^{-2t} tr(\left(C^0\right)^{-1} - \Lambda), \end{aligned}$$

where m^t and C^t are the empirical mean and covariance matrix at time t and $\exp(-\Lambda t)$ is the matrix exponential.

Proof: The proof is given in Appendix F.

Our result shows that convergence of the mean m^t depends on the eigenvalues of Λ. From our partial result for the covariance C^t we might conjecture that convergence of C^t could be (asymptotically) independent of the target parameters. We confirm this behaviour

Figure 1: Inference on a multivariate Gaussian target distribution with random mean and a) isotropic covariance b) full-rank covariance. We compare GPF and GDP with SVGD and GVA in terms of one standard deviation around wall-clock time, distance to the true mean and the covariance matrix (y-axis) as the dimensionality (x-axis) of the Gaussian target increases. We use $D + 1$ particles/samples.



a) Isotropic covariance $\Sigma = I$. b) $\Sigma = U\Lambda U^\top$ where U is unitary and $\Lambda_{ii} = 10^{\frac{3i-1}{D}}$.

for Gaussian targets in the experiments in Section 3. We also show that good approximation can be obtained for a moderate number of particles. If the number of particles is in fact equal or smaller than the dimension, i.e. $N < D + 1$, we can show the following result for the fixed points:

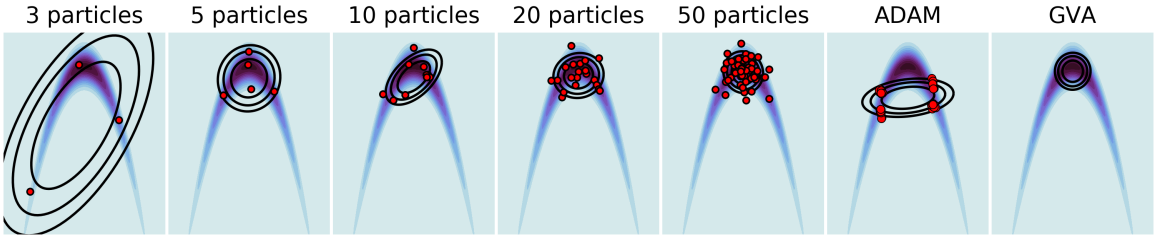
Theorem 3 *Given a D-dimensional multivariate Gaussian target density $p(x) = \mathcal{N}(x|\mu, \Sigma)$, using Algorithm 1 with $N < D + 1$ particles, the empirical mean converges to the exact mean μ. The $N - 1$ non-zero eigenvalues of C^t converge to a subset of the spectrum of the target covariance Σ.*

Proof: The proof is given in Appendix G. Supported by simulations, we conjecture that we recover in fact the largest eigenvalues of Σ in this way. This results suggests that it makes sense to apply our algorithm to high–dimensional problems even when the number of particles is not large. If the target density has significant support in a low dimensional sub–manifold only, we might still get a reasonable approximation.

3. Experiments

We now demonstrate empirically the benefits of GPF. First, we explore how the different algorithm hyperparameters affect inference. Second, we compare the ability of GPF to recover the target posterior distribution with popular VI methods, including SVGD ands GVA. In particular, we work fully-controlled synthetic examples based on Gaussian target distributions with an increasing number of dimensions as well as highly non-Gaussian posteriors. We evaluate the ability of GPF to recover the ground truth as the number of particles increases and compare it to competing VI approaches. We use standard gradient descent with $\eta = 0.1$ as well as a clipping on the gradient norm with a threshold $\epsilon = 10$ in all our experiments. We run each experiment for 2000 iterations and aggregate results over 10 independent runs. We also evaluate the efficiency of our algorithm on *Gaussian Processes* **(GP)**s and *Bayesian Neural Networks* **(BNN)**s in Appendix H.

Figure 2: Two-dimensional Banana distribution. GPF with an increasing number of particles compared to a different optimizer (ADAM) and standard GVA.



Gaussian Targets Consider a D-dimensional Gaussian target. The mean is sampled from a normal Gaussian $\mu \sim \mathcal{N}(0, I_D)$ and we consider two different cases for the covariance: isotropic $\Sigma = I_D$ and full-rank covariance $\Sigma = U\Lambda U^\top$, where U is a unitary matrix and Λ is a diagonal matrix where $\Lambda_{ii} = 10^{\frac{3i-1}{D}-1}$. We compare GPF and GDP to two state-of-the art methods described in Appendix A, namely SVGD and GVA. We vary the number of dimensions D and set the number of particles or alternatively the number of samples for GVA to $D+1$. Figure 2.4 reports the L_2 norm on the difference of mean and covariance with the true posterior, and the inference time required by the competing methods. GPF recovers the true mean and covariance at a fraction of the time required by SVGD and GVA. The gap is more pronounced as the target dimensionality increases.

Non-Gaussian target distributions and optimizers We investigate non-Gaussian and multi-modal target distributions. We compare different optimizers on the 2D banana distribution: $p(x) \propto \exp\left(-0.5(0.01x_1^2 + 0.1(x_2 + 0.1x_1^2 - 10)^2)\right)$. We vary the number of particles used for inference in $\{3, 5, 10, 20, 50\}$ and compare GPF with the standard GVA approach. We also show the impact of replacing SGD with Adam (Kingma and Ba, 2014) for 50 particles. The results are shown in Figure 3. As expected, increasing the number of particles makes the distribution obtained via GPF increasingly closer to the one obtained by the standard GVA at much lower computational cost. Using a momentum based optimizer such as Adam breaks the Gaussian distribution and particles concentrate on two modes. This is caused by momentum-based optimizers breaking the linearity of Equation (6), which could studied as a desirable property to fit multi-modal distributions in future work.

4. Conclusion

We introduced GPF and GDP, general-purpose, scalable, and theoretically-grounded approaches to perform VI. The key contribution is to leverage deterministic dynamics to perform tractable GVA in high dimensions, making it applicable to a broad set of challenging, real-world high-dimensional problems. We also derived theoretical performance guarantees for Gaussian targets.We studied the methods on synthetic data as well as more challenging problems. Investigations are still needed to evaluate the effect of the empirical estimate for the free energy, especially in the regime of a small number of particles. Another promising direction is the effect of momentum-based optimizers to obtain multi-modal variational distributions.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Amari, S. I. (1998). Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10(2):251–276. ZSCC: 0002989 ISBN: 0899-7667.

Bardenet, R., Doucet, A., and Holmes, C. (2017). On markov chain monte carlo methods for tall data. *The Journal of Machine Learning Research*, 18(1):1515–1557.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877.

Cacoullos, T. (1964). Estimation of a multivariate density. Technical report, University of Minnesota.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. cite arxiv:1512.01274Comment: In Neural Information Processing Systems, Workshop on Machine Learning Systems, 2016.

Cowles, M. K. and Carlin, B. P. (1996). Markov chain monte carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, 91(434):883–904.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Ehrendorfer, M. (2007). A review of issues in ensemble-based kalman filtering. *Meteorologische Zeitschrift*, 16(6):795–818.

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. *Journal of Geophysical Research: Oceans*, 99(C5):10143–10162.

Fix, E. and Hodges, J. (1951). Nonparametric discrimination: Consistency properties. *Randolph Field, Texas, Project*, pages 21–49.

Gal, Y. and Ghahramani, Z. (2015). Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*.

Genevay, A., Cuturi, M., Peyré, G., and Bach, F. (2016). Stochastic optimization for large-scale optimal transport. In *Advances in neural information processing systems*, pages 3440–3448.

Gershman, S., Hoffman, M., and Blei, D. (2012). Nonparametric variational inference. *arXiv preprint arXiv:1206.4665*.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. *arXiv preprint arXiv:1706.04599*.

Hensman, J., Matthews, A., and Ghahramani, Z. (2015). Scalable variational gaussian process classification. *The Journal of Machine Learning Research*.

Hinton, G. E. and van Camp, D. (1993). Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, page 5–13, New York, NY, USA. Association for Computing Machinery.

Ingersoll, J. E. (1987). *Theory of financial decision making*, volume 3. Rowman & Littlefield.

Innes, M. (2018). Flux: Elegant machine learning with julia. *Journal of Open Source Software*.

Khan, M. E. and Nielsen, D. (2018). Fast yet simple natural-gradient descent for variational inference in complex models. In *2018 International Symposium on Information Theory and Its Applications (ISITA)*, pages 31–35. IEEE.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

LeCun, Y. and Cortes, C. (2010). MNIST handwritten digit database.

Lin, W., Khan, M. E., and Schmidt, M. (2019). Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. *arXiv preprint arXiv:1906.02914*.

Liu, C., Zhuo, J., Cheng, P., Zhang, R., and Zhu, J. (2019). Understanding and accelerating particle-based variational inference. In *International Conference on Machine Learning*, pages 4082–4092.

Liu, Q., Lee, J., and Jordan, M. (2016). A kernelized stein discrepancy for goodness-of-fit tests. In *International conference on machine learning*, pages 276–284.

Liu, Q. and Wang, D. (2016). Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in neural information processing systems*, pages 2378–2386.

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., and Wilson, A. G. (2019). A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13153–13164.

Opper, M. and Archambeau, C. (2009). The variational gaussian approximation revisited. *Neural computation*, 21(3):786–792.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Ranganath, R., Gerrish, S., and Blei, D. (2014). Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822. PMLR.

Rezende, D. J. and Mohamed, S. (2015). Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*.

Saeedi, A., Kulkarni, T. D., Mansinghka, V. K., and Gershman, S. J. (2017). Variational particle approximations. *The Journal of Machine Learning Research*, 18(1):2328–2356.

Settles, B. (2009). Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175.

Shridhar, K., Laumann, F., and Liwicki, M. (2019). A comprehensive guide to bayesian convolutional neural network with variational inference. *arXiv preprint arXiv:1901.02731*.

Sigillito, V. G., Wing, S. P., Hutton, L. V., and Baker, K. B. (1989). Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, 10(3):262–266.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. The MIT Press.

Titsias, M. and Lázaro-Gredilla, M. (2014). Doubly stochastic variational bayes for non-conjugate inference. In *International conference on machine learning*, pages 1971–1979.

Wenzel, F., Galy-Fajou, T., Donner, C., Kloft, M., and Opper, M. (2019). Efficient gaussian process classification using pòlya-gamma data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5417–5424.

Zhu, M. H., Liu, C., and Zhu, J. (2020). Variance reduction and quasi-newton for particle-based variational inference. In *International Conference on Machine Learning*.

Zhuo, J., Liu, C., Shi, J., Zhu, J., Chen, N., and Zhang, B. (2018). Message passing stein variational gradient descent. volume 80, pages 6018–6027.

Appendix A. Related Work

A diverse set of approaches to perform VI have been developed in the literature, which we review here.

High-dimensional VI The most popular approach to solve high-dimensional problems via variational inference is (structured) *Mean-Field* **(MF)** (Hinton and van Camp, 1993; Blei et al., 2017). By imposing independence between variables in the variational distribution, the computational complexity decreases at the cost of the approximation quality. For Gaussian distributions the most common approach is to leverage the *Gaussian Variational Approximation* **(GVA)** (Opper and Archambeau, 2009), where the expectation gradients given the variational parameters can be obtained directly with a Monte-Carlo estimation. More recently Titsias and Lázaro-Gredilla (2014) avoid the need for integrals by proceeding to stochastic updates based on the idea of *Stochastic Gradient Descent* **(SGD)**. The most generic technique is *Black-Box Variational Inference* (Ranganath et al., 2014; Blei et al., 2017), which is based on the *reparametrization trick*. This avoids unwanted dependencies of the sampling distribution on parameters and yields unbiased estimates of parameter gradients. However, it is nontrivial to apply this idea to a full rank D-dimensional multivariate variational Gaussian density. One possibility would be to resort to Cholesky factorizations of the covariance matrix in the parametrisation. This would scale with a prohibitive $\mathcal{O}(D^3)$ complexity. In addition, efficient updates based on natural gradient descent methods (Khan and Nielsen, 2018; Lin et al., 2019) require extra second derivatives of log–posteriors. These derivatives may not be always available in a black-box scenario. Hence, extra approximations using outer products of gradients are used instead.

Particle-Based VI Stochastic gradient descent methods compute expectations (and gradients) with Monte Carlo samples from the current approximation of the variational density independently at each time step. Particle based methods for variational inference *draw samples once* at the beginning of the algorithm instead. They iteratively construct a transformation on an initial random variable (with a simple tractable density) leading to the minimization of the variational free energy. The iterative approach induces a deterministic temporal flow of random variables which depends on the current density of the variable itself. Using an approximation by the empirical density (which is represented by a the positions of a set of "particles") one obtains an interacting particle flow which converges asymptotically to an empirical approximation of the desired optimal variational density.

The most popular approach is SVGD (Liu and Wang, 2016), which finds a nonparametric transport map transforming a collection of particles into samples from the target distribution based on the kernelized Stein discrepancy (Liu et al., 2016). SVGD has the advantage of not being restricted to a parametric form of the variational distribution. On the other hand, useful features such as an explicit expression for the variational free energy or structured approximations, such as a mean-field method, are not yet available. A mean field approximation could have a regularising effect for finite sample sizes in high dimensions. Other limitations of SVGD are the quadratic scaling with the number of samples and the approximation of the kernelized Stein discrepancy. This is known to not perform well in high dimensions (Zhuo et al., 2018).

Non-parametric Variational Inference presents a different approach where each particle represents a Gaussian (similarly to *Kernel Density Estimation* in Fix and Hodges (1951) and Cacoullos (1964) with a Gaussian kernel) and the KL-divergence between the mixture of Gaussians and the target posterior is minimized (Gershman et al., 2012). However, due to the complicated behavior of Gaussian densities in high-dimensions, this method does not scale well. *Variational Particle Approximation* (Saeedi et al., 2017) also use particles to represent a variational distribution, but the method is only applicable to discrete variables.

Related approaches The closest approach to our proposed method is *Ensemble Kalman Filter* (Evensen, 1994), where a Gaussian distribution is repeatedly approximated by a set of particles, iteratively moved following the Kalman filter updates. However, this method is typically applied to an online context, which poses problems that do not hold in our variational setting (Ehrendorfer, 2007). A related method recently introduced in the context of BNN models is *Stochastic Weight Averaging - Gaussian* **(SWAG)** (Maddox et al., 2019), in which a set of particles obtained via stochastic gradient descent represent a low-rank Gaussian distribution, approximating the true posterior with a prior produced by the network's regularization. While easy to implement, SWAG does not allow for incorporating an explicit prior. Finally, we mention *Normalizing Flows* **(NF)** (Rezende and Mohamed, 2015), where bijections are sequentially stacked on top of a fixed initial distribution. The repeated transformations on an initial distribution can be viewed as a discrete flow. This is orthogonal to our approach and could be combined with GPF by applying NF on the variational distribution. For instance, this makes it possible to work with different domains than $\mathbb{R}$.

Appendix B. Natural gradient for matrix parameter Γ

The natural gradient is defined as the parameter change $d\Gamma = \Gamma(t + dt) - \Gamma(t)$ over a small time interval dt which yields the direction of steepest descent of the free energy. As an extra condition one keeps the length of $d\Gamma$ (measured by a 'natural' metric which has specific invariance properties) fixed. The metric is defined by an inner product (the squared length) $\langle d\Gamma, d\Gamma\rangle_\Gamma$ in the tangent space of small deviations $d\Gamma$ from the matrix Γ. Hence, $d\Gamma$ is found by minimising $\mathcal{F}(\Gamma(t) + d\Gamma, m)$ (for small $d\Gamma$) under the condition that $\langle d\Gamma, d\Gamma\rangle_{\Gamma(t)}$ is fixed. Following Amari (1998, 7.6) a natural metric in the space of symmetric nonsingular matrices can be defined as

$$\langle d\Gamma, d\Gamma\rangle_\Gamma \doteq \mathrm{tr}\left((d\Gamma\ \Gamma^{-1})^\top d\Gamma\ \Gamma^{-1}\right).$$

This metric is invariant against multiplications of Γ and $d\Gamma$ by matrices Y, i.e. $\langle d\Gamma, d\Gamma\rangle_\Gamma = \langle d\Gamma\ Y, d\Gamma\ Y\rangle_{\Gamma Y}$ and reduces to the Euclidian metric at the unit matrix $\Gamma = I$.

The direction of the natural gradient is obtained by expanding the free energy for small $d\Gamma$ and introducing a Lagrange–multiplier λ for the constraint. One ends up with the quadratic form

$$\mathrm{tr}\left(\frac{\partial \mathcal{F}}{\partial \Gamma}^\top d\Gamma\right) + \lambda \mathrm{tr}\left((d\Gamma\ \Gamma^{-1})^\top d\Gamma\ \Gamma^{-1}\right)$$

to be minimised by $d\Gamma$. By taking the derivative with respect to $d\Gamma$ one finds that the direction of $d\Gamma$ agrees with the right equation of the flow (4).

Appendix C. Free Energy as a function of particles

In Equation (3) we have the free energy $\mathcal{F}$ as a function of parameters Γ and m. Here instead, we only consider the random variable $x \sim q(x)$, as well as its mean $m = \mathbb{E}_q[x]$ and its covariance $C = \mathbb{E}_q\left[(x-m)(x-m)^\top\right]$. The free energy as a function of x is given by

$$\mathcal{F}(x) = \mathbb{E}_q[\varphi(x)] - \frac{1}{2}\log|C|.$$

Similarly to the derivations in Section 2, $q(x)$ is either a Gaussian distribution $q(x) \sim \mathcal{N}(m, C)$ or an empirical distribution $\hat{q}(x) = \frac{1}{N}\sum_{i=1}^N \delta_{x,x_i}$ with a finite number of particles.

We now consider time dynamics on x, i.e. $x = x(t)$. The induced dynamics of $\mathcal{F}$ are:

$$\frac{d\mathcal{F}}{dt} = \mathbb{E}_q\left[\nabla_x\varphi(x)^\top \frac{dx}{dt}\right] - \frac{1}{2}\mathrm{tr}(C^{-1}\frac{dC}{dt})$$

For notation simplicity we define $g(x) = \nabla_x\varphi(x)$ and $\dot{x} = \frac{dx}{dt}$ (similarly $\dot{m} = \frac{dm}{dt}$).

$$\begin{aligned}
\frac{dC}{dt} =& \frac{d}{dt}\mathbb{E}_q\left[(x-m)(x-m)^\top\right] \\
=& \mathbb{E}_q\left[(\dot{x}-\dot{m})(x-m)^\top\right] + \mathbb{E}_q\left[(x-m)(\dot{x}-\dot{m})^\top\right] \\
=& \mathbb{E}_q\left[\dot{x}x^\top x\dot{x}^\top - \dot{m}m^\top - m\dot{m}^\top\right] \\
=& \mathbb{E}_q\left[\dot{x}(x-m)^\top\right] + \mathbb{E}_q\left[(x-m)\dot{x}^\top\right]
\end{aligned}$$

$$\begin{aligned}
\frac{d\mathcal{F}}{dt} =& \mathbb{E}_q\left[g(x)^\top\dot{x}\right] - \\
& \frac{1}{2}\mathbb{E}_q\left[\mathrm{tr}(C^{-1}\dot{x}(x-m)^\top) + \mathrm{tr}(C^{-1}(x-m)^\top\dot{x}^\top)\right] \\
=& \mathbb{E}_q\left[\dot{x}^\top\left(g(x) - C^{-1}(x-m)\right)\right]
\end{aligned} \tag{9}$$

where we used the permutation properties of the trace. We now study the effect of linear dynamics on x, i.e.:

$$\dot{x} = b + A(x-m).$$

Plugging those dynamics into Equation (9), we get:

$$\begin{aligned}
\frac{d\mathcal{F}}{dt} =& b^\top\mathbb{E}_q[g(x)] + \mathbb{E}_q\left[(x-m)^\top A^\top g(x)\right] \\
& - \mathbb{E}_q\left[(x-m)^\top A^\top C^{-1}(x-m)\right]
\end{aligned} \tag{10}$$

where we ignored the term $b^\top C^{-1}\mathbb{E}_q[x-m]$ as it is naturally equal to 0.

We now find terms for b and A such that $\frac{d\mathcal{F}}{dt} < 0$, i.e. the dynamics will lead to a decrease of the free energy. We pick $b = -\beta_1 \mathbb{E}_q\left[g(x)\right]$, where $\beta_1 > 0$, and obtain a negative first term:

$$-\beta_1 \|\mathbb{E}_q\left[g(x)\right]\|^2 \leq 0.$$

For A let's first define $\psi = \mathbb{E}_q\left[g(x)(x-m)^\top\right]$ and rewrite the second and last term of the Equation 10 as:

$$\begin{aligned}
\mathbb{E}_q\left[(x-m)^\top A^\top g(x)\right] &= \mathrm{tr}\left(\mathbb{E}_q\left[A^\top g(x)(x-m)^\top\right]\right) \\
&= \mathrm{tr}\left(A^\top \psi\right) \\
\mathbb{E}_q\left[(x-m)^\top A^\top C^{-1}(x-m)\right] &= \mathrm{tr}\left(A^\top C^{-1} C\right) \\
&= \mathrm{tr}(A)
\end{aligned}$$

Combining both we get: $\mathrm{tr}\left(A^\top(\psi - I)\right)$. Similarly to the previous step we pick $A = -\beta_2\left(\psi - I\right)$, where $\beta_2 \geq 0$, which leads to another negative term:

$$-\beta_2 \mathrm{tr}((\psi - I)^\top(\psi - I)) \leq 0,$$

where we use the fact that $X^\top X$ is a positive semi-definite matrix for any real valued X and that its eigenvalues are all greater or equal to 0.

Note that a different A could be used as long as the trace of the product stays positive. Replacing b and A, the free energy dynamics become:

$$\frac{d\mathcal{F}}{dt} = -\beta_1 \|\mathbb{E}_q\left[g(x)\right]\|^2 - \beta_2 \mathrm{tr}((\psi - I)^\top(\psi - I))$$

And the variable dynamics are:

$$\begin{aligned}
\frac{dx}{dt} &= -\beta_1 \mathbb{E}_q\left[g(x)\right] - \beta_2\left(\psi - I\right)(x-m) \\
&= -\beta_1 \mathbb{E}_q\left[g(x)\right] \\
&\quad - \beta_2\left(\mathbb{E}_q\left[g(x)(x-m)^\top\right] - I\right)(x-m),
\end{aligned}$$

which is equivalent to Equation (6), for $\beta_1 = \beta_2 = 1$. As in Equation (7), when considering a finite number of samples, we use instead an empirical approximation of the free energy and replace the expectations by empirical estimates.

Appendix D. Algorithms

Appendix E. Fixed points for a Gaussian model ($N > d$)

The general fixed-point condition for the dynamics (7) of the position x_i for particle i is given by

$$(I - \mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right])(x_i - m) - \mathbb{E}_{\hat{q}}\left[g(x)\right] = 0.$$

Input: Number of particles N, initial distribution q_0, target $p(x)$, learning rates η_1, η_2
Output: Empirical dist. $q(x) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x,x_i}$
Init: Sample N particles from q_0: $\{x_i\}_{i=1}^{N}$
for j *in* $1:T$ **do**
  Compute gradients $g_i = \nabla_x \log p(x_i),\ \forall i$
  Compute means $m = \frac{1}{N}\sum_i x_i, \quad \bar{g} = \frac{1}{N}\sum_i g_i$
  Compute matrix $A = \frac{1}{N}\sum_i g_i (x_i - m)^\top - I$
  Update particles $x_i = -\eta_1 \bar{g} - \eta_2 A(x_i - m),\ \forall i$
end

Algorithm 1: Gaussian Particle Flow (GPF)

Input: Number of samples N, initial parameters m, Γ, target $p(x)$, learning rates η_1, η_2
Output: Gaussian distribution $q(x) = \mathcal{N}\left(m, \Gamma\Gamma^\top\right)$
for j *in* $1:T$ **do**
  Sample N samples from q_0: $\{x_i^0\}_{i=1}^{N}$
  Project the samples using m and Γ: $x_i = m + \Gamma(x_i^0 - m^0)$
  Compute gradients $g_i = \nabla_x \log p(x_i),\ \forall i$
  Compute means $m = \frac{1}{N}\sum_i x_i, \quad \bar{g} = \frac{1}{N}\sum_i g_i$
  Compute matrix $A = \frac{1}{N}\sum_i g_i (x_i - m)^\top - I$
  Update parameters $m \leftarrow m - \eta_1 \bar{g}, \quad \Gamma \leftarrow \Gamma - \eta_2 A\Gamma,\ \forall i$
end

Algorithm 2: Gaussian Density Parametrization (GDP)

for $i = 1, \ldots, N$. By taking the expectation over all particles we get:

$$\mathbb{E}_{\hat{q}}\left[g(x)\right] = 0, \tag{11}$$

where $\hat{q}$ is the empirical distributions of particles at the the fixed point. Note that this result is independent of N, i.e. it is also valid for $N = 1$.

For a D-dimensional Gaussian target, i.e. $p(x) = \mathcal{N}(\mu, \Sigma)$, we will show that empirical mean and covariance given by the particle algorithm converge to the true mean and covariance matrix of the Gaussian when we use $N \geq D + 1$ particles. In this setting we have $\varphi(x) = \frac{1}{2}x^\top \Sigma^{-1} x - x^\top \Sigma^{-1}\mu$. For simplification we use natural parameters $\eta_1 = \Sigma^{-1}\mu$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$ and get

$$\varphi(x) = -x^\top \eta_2 x - x^\top \eta_1.$$

The gradient $g(x)$ becomes:

$$g(x) = -2\eta_2 x - \eta_1$$

At the fixed points we have that $\frac{dm}{dt}$ and $\frac{d\Gamma}{dt}$ are equal to 0. For the mean m:

$$\begin{aligned}
\frac{dm}{dt} = -\mathbb{E}_{\hat{q}}\left[g(x)\right] &= 0 \\
2\eta_2 \mathbb{E}_{\hat{q}}\left[x\right] + \eta_1 &= 0 \\
2\eta_2 m + \eta_1 &= 0 \\
m &= -\frac{1}{2}\eta_2^{-1}\eta_1 \\
m &= \mu
\end{aligned}$$

For the matrix Γ:

$$\begin{aligned}
\frac{d\Gamma}{dt} = -A\Gamma &= 0 \\
\Gamma - \mathbb{E}_{q_0}\left[g(x)(x-m)^\top\right]\Gamma &= 0 \\
\mathbb{E}_{q_0}\left[-(2\eta_2 x + \eta_1)(x-m)^\top\right]\Gamma &= \Gamma \\
\mathbb{E}_{q_0}\left[-2(\eta_2 x - \eta_2 m)(x-m)^\top\right]\Gamma &= \Gamma \\
-2\eta_2 \mathbb{E}_{q_0}\left[(x-m)(x-m)^\top\right]\Gamma &= \Gamma \\
-2\eta_2 C\Gamma &= \Gamma \\
-2\eta_2 C^2 &= C
\end{aligned}$$

Where we replaced η_1 by $-2\eta_2 m$ given the previous result and right multiplied by $\Gamma^\top$ as $C = \Gamma\Gamma^\top$. Now we can only simplify as $C = -\frac{1}{2}\eta_2^{-1} = \Sigma$ iff C is not singular. Which is true only if its rank is equal to D, needing $D+1$ particles.

Appendix F. Rates of Convergence for Gaussian model

F.1. Convergence of the mean

Given a Gaussian target $p(x) = \mathcal{N}(x|\mu, \Sigma)$, as described in Appendix E, we have $g(x) = -2\eta_2 x - \eta_1$, where $\eta_1 = \Sigma^{-1}\mu$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$. This transform the of Equations (5) into:

$$\begin{aligned}
\frac{dm}{dt} &= -2\eta_2 \mathbb{E}_{\hat{q}}\left[x\right] - \eta_1 \\
&= -2\eta_2 m - \eta_1
\end{aligned}$$

If now consider the error on m: $\delta m = m - \mu$ we get:

$$\begin{aligned}\frac{d\delta m}{dt} =& \frac{dm}{dt} = -2\eta_2 m - \eta_1 \\ =& -2\eta_2(\delta m + \mu) - \eta_1 \\ =& -2\eta_2(\delta m - \frac{1}{2}\eta_2^{-1}\eta_1) \\ =& -2\eta_2 \delta m \\ =& \Sigma^{-1}\delta m.\end{aligned}$$

Therefore we have a linear convergence which is asymptotically governed by the largest eigenvalue of Σ^{-1}, i.e. the inverse of the smallest eigenvalue of Σ, $\lambda_{\min}$.

F.2. Convergence of the covariance matrix

For the convergence of the covariance we focus once more on the Gaussian case. Let $z = x - m$, we have that from Equation (6):

$$\frac{dz}{dt} = -Az$$

where $A = \mathbb{E}_{q_0}\left[g(x)z^\top\right] - I$. This expectation can further simplified as:

$$\begin{aligned}\mathbb{E}_{\hat{q}}\left[(-2\eta_2 x - \eta_1)z^\top\right] =& -2\eta_2 \mathbb{E}_{\hat{q}}\left[xz^\top\right] \\ =& -2\eta_2 \mathbb{E}_{\hat{q}}\left[(x-m)z^\top\right] \\ & -2\eta_2 m \underbrace{\mathbb{E}_{\hat{q}}\left[z\right]}_{:=0} \\ =& -2\eta_2 C,\end{aligned} \tag{12}$$

where $q \sim \mathcal{N}(m, C)$. Hence we have the exact result:

$$\frac{dC}{dt} = (2\eta_2 C + I)C + C(2C\eta_2 + I). \tag{13}$$

We know that the optimal target is $C = -\frac{1}{2}\eta_2^{-1}$. Therefore we define the error $\delta C = C + \frac{1}{2}\eta_2^{-1}$. Linearizing Equation (13) gives us:

$$\begin{aligned}\frac{d\delta C}{dt} =& \frac{dC}{dt} + \underbrace{\frac{1}{2}\frac{d\eta_2^{-1}}{dt}}_{:=0} \\ =& (2\eta_2(\delta C - \frac{1}{2}\eta_2^{-1}) + I)(\delta C - \frac{1}{2}\eta_2^{-1}) \\ & + (\delta C - \frac{1}{2}\eta_2^{-1})(2(\delta C - \frac{1}{2}\eta_2^{-1})\eta_2 + I) \\ =& 2\eta_2 \delta C(\delta C - \frac{1}{2}\eta_2^{-1}) + 2(\delta C - \frac{1}{2}\eta_2^{-1})\delta C\eta_2 \\ \approx& -\eta_2 \delta C \eta_2^{-1} - \eta_2^{-1}\delta C \eta_2\end{aligned}$$

We were not yet able to find a general solution of this equation, but we can obtain a simple result for the trace $y^t \doteq \mathrm{tr}(\delta C)$ at time t:

$$\frac{dy^t}{dt} \simeq -2y^t$$

We therefore have a asymptotic linear convergence: $y^t \propto e^{-2t}y^0$ which is independent of the parameters of the Gaussian model.

We can also get an non-asymptotic estimate of a specific error measure for the precision matrix. Using equation (13), we have the following dynamics for the precision C^{-1}:

$$\begin{aligned}\frac{dC^{-1}}{dt} =& -C^{-1}\frac{dC}{dt}C^{-1} \\ =& -C^{-1}(I + 2\eta_2 C) - (I + 2\eta_2 C)C^{-1}\end{aligned}$$

Taking the trace

$$\begin{aligned}\frac{d\mathrm{tr}(C^{-1})}{dt} =& -2\mathrm{tr}(C^{-1}) - 4\mathrm{tr}(\eta_2) \\ =& -2\mathrm{tr}(C^{-1}) - 2\mathrm{tr}(\Sigma^{-1}) \\ \frac{d\mathrm{tr}(C^{-1} - \Sigma^{-1})}{dt} =& -2\mathrm{tr}(C^{-1} - \Sigma^{-1})\end{aligned}$$

Hence we get the following exact result:

$$\mathrm{tr}((C^t)^{-1} - \Sigma^{-1}) = e^{-2t}\mathrm{tr}((C^0)^{-1} - \Sigma^{-1})$$

which is again independent of the parameters of the Gaussian model.

Additionally this tells us that if the covariance C is non-singular at time $t = 0$, it will remain non-singular for all t ($\mathrm{tr}(C^{-1})$ would be infinite). Hence if we start with $N > d$ particles with a proper empirical covariance, they cannot collapse to make C singular

Appendix G. Fixed-points for Gaussian model ($N \leq D$)

Applying equation (11) to our fixed point equation we get

$$(I - \mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right])(x_i - m) = 0,\ \forall i = 1, \ldots, N$$

Hence, the set of centered positions of the particles $S = \{x_i - m\}_{i=1}^N$, are all eigenvectors of the matrix $\mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right]$ with eigenvalue 1. S spans a $N-1$ dimensional space (we have $\sum_{i=1}^N (x_i - m) = 0$).

If we specialise to a Gaussian target $p(x) = \mathcal{N}(x \mid \mu, \Sigma)$, we have $g(x) = -2\eta_2 x - \eta_1$ and can reuse the result from Equation (12):

$$\begin{aligned}\mathbb{E}_{\hat{q}}\left[g(x)(x-m)^\top\right] =& -2\eta_2 \mathbb{E}_{\hat{q}}\left[(x-m)(x-m)^\top\right] \\ =& -2\eta_2 C \\ \Sigma^{-1}C(x_i - m) =& (x_i - m) \\ C(x_i - m) =& \Sigma(x_i - m)\end{aligned}$$

which shows that the obtained low-rank covariance C and the target covariance Σ have $N-1$ eigenvectors and eigenvalues in common.

Appendix H. Additional experiments

H.1. Gaussian Process Classification

We turn to estimating the quality of the variational distribution given a varying number of particles for a non-Gaussian posterior. Specifically, we consider a GP classifier: $y \sim \text{Bernoulli}(\sigma(f))$, where σ is the logistic function on the Ionosphere dataset (Sigillito et al., 1989) from the UCI repository (Dua and Graff, 2017). This dataset contains 351 instances and 34 features and is split into a train/test set with 231/120 instances. The dimensionality of a GP is equal to the number of training instances. However, our method does not decrease the cubic complexity, as the GP prior computations are still the bottleneck. We sample from the posterior using the Gibbs sampling approach from Wenzel et al. (2019) and train a VI model using the method described in Hensman et al. (2015). For all experiments the kernel parameters are fixed during training. We train GPF starting with a MAP estimation and we estimate the predictions on a test . To compare the accuracy of the results, we compute the Wasserstein distance with the L_2 norm, using the stochastic approach from Genevay et al. (2016). The Wasserstein metric has the advantage of allowing us to compare both particle-based method and parameter-based methods. Figure H.1 compares accuracy, negative log-likelihood and the Wasserstein metrics of GPF with *Markov chain Monte Carlo* **(MCMC)** (used as the ground truth). The results show that GPF is able to match the accuracy results of VI for a sufficiently large number of particles. As expected, with a number of particles that closely matches the problem dimensionality (i.e. 231 the training set size), GPF closely matches the performance of VI. Moreover we see that an increasing number of particles progressively reduces the Wasserstein distance between our variational distribution and the ground truth.

H.2. Bayesian Neural Networks

Finally, we evaluate our method on the challenging problem of inference with high-dimensional *Bayesian Neural Networks* **(BNN)**. We consider the `LeNet` convolutional neural network (LeCun et al., 1998) trained on `MNIST` (LeCun and Cortes, 2010) implemented in Flux.jl Innes (2018). We first train `LeNet` until we get the maximum likelihood (**ML**) estimate; from there, we fix the weights of the convolutional layers and focus on estimating the weights of the dense layers (the weights of the convolutional layers are notoriously difficult to infer; see Gal and Ghahramani (2015) and Shridhar et al. (2019)). The problem is thus very high-dimensional, with 41,854 parameters (i.e., the number of weights in the dense layers). We compare GPF to SWAG (Maddox et al., 2019) and GVA. For SWAG we do not place a prior on the weights, use the momentum optimizer with constant learning rate ($\eta = 0.05$) and use the last 50 samples taken every 4th of an epoch. For GPF we use 50 particles and consider two MF variants: *Structured MF* by assuming independence between layers and *MF* by assuming independence between each weight. For GVA we use

Figure 3: Gaussian Process Classification. We compute the average test accuracy and negative log-likelihood of GPF for a varying number of particles, as well as the train Wasserstein distance W_2 (mean and one standard deviation over 10 runs). We compare it with the true posterior computed via sampling (MCMC) and the optimal VI result. Increasing the number of particles to match the problem dimensionality (vertical black line) progressively allows GPF to reach the optimal value of the standard VI approach.

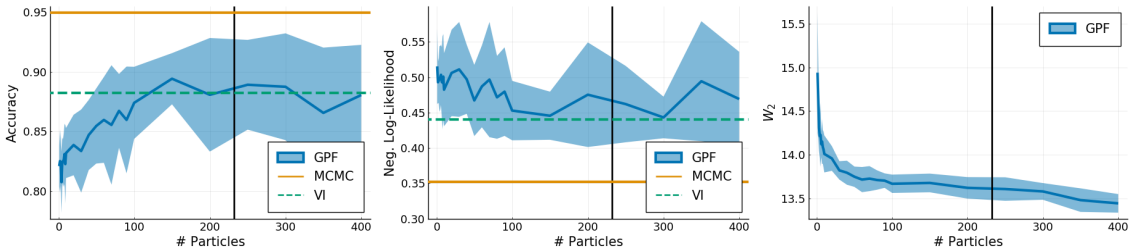


Figure 4: Bayesian neural network (LeNet) trained on MNIST. Comparison of GPF (both with full MF and structureed MF) with SWAG and the GVA in terms of the calibration of the predictions.

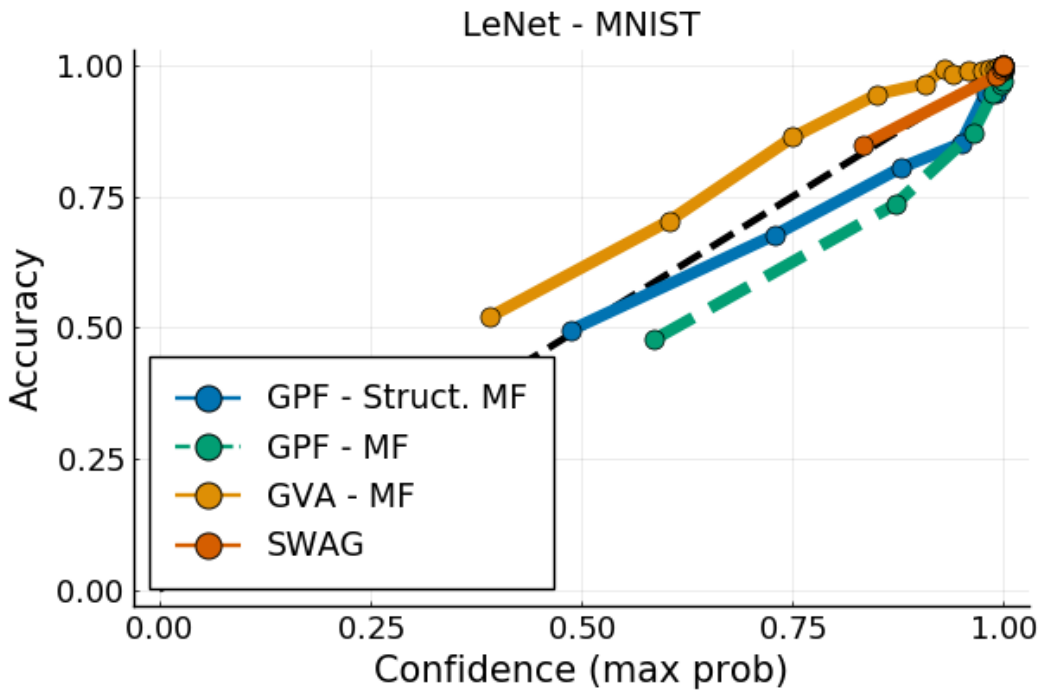
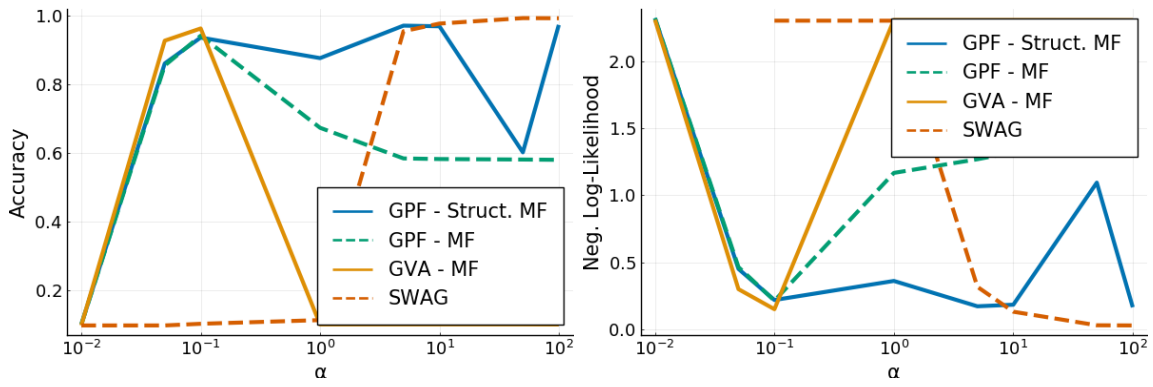


Figure 5: Accuracy and negative Log-Likelihood as a function of α, the standard deviation of the weights prior. We compare GPF with GVA and SWAG.



50 samples for each expectation estimation and assume independence between weights, as even a structured approach would be prohibitively expensive.

We compare different models in terms of standard performance metrics, namely negative log-likelihood and accuracy, as well as in terms of calibration. To compute the calibration we use the technique from Guo et al. (2017): we take the maximum probability among all classes and split them into 20 bins of equal frequency. We then compute the accuracy of the samples contained in each bin. In a perfectly calibrated model, the obtained probability and the accuracy should be the same. Figure H.1 compares the competing approaches in terms of the calibration of the predictions, showing that GPF achieves considerably better performance than GVA, and perform similarly to ML and SWAG. Figure H.1 compares the averaged negative log-likelihood and the accuracy when varying the value of the prior standard deviation α. We can also observe that having a structured MF approach (even low-rank) improves both on the calibration and on the quality of the obtained posterior. SWAG generally outperforms competitors but is also less flexible and less principled. Unlike GPF, it can only be used in the context of SGD and cannot be applied to tackle general Bayesian inference problems. Interestingly, despite being a general purpose method GPF is able to achieve on par calibration with specialized techniques.