Efficient Gradient-Free Variational Inference using Policy Search

Oleg Arenz [1] **Mingjun Zhong** [2] **Gerhard Neumann** [1 3]

Abstract

Inference from complex distributions is a common problem in machine learning needed for many Bayesian methods. We propose an efficient, gradient-free method for learning general GMM approximations of multimodal distributions based on recent insights from stochastic search methods. Our method establishes information-geometric trust regions to ensure efficient exploration of the sampling space and stability of the GMM updates, allowing for efficient estimation of multi-variate Gaussian variational distributions. For GMMs, we apply a variational lower bound to decompose the learning objective into sub-problems given by learning the individual mixture components and the coefficients. The number of mixture components is adapted online in order to allow for arbitrary exact approximations. We demonstrate on several domains that we can learn significantly better approximations than competing variational inference methods and that the quality of samples drawn from our approximations is on par with samples created by state-of-the-art MCMC samplers that require significantly more computational resources.

1. Introduction

We consider the problem of sampling or inference using a complex probability distribution $p^*(\boldsymbol{x}) = \tilde{p}^*(\boldsymbol{x})/Z$ for which we can evaluate $\tilde{p}^*(\boldsymbol{x})$ but not the normalization constant $Z = \int_{\boldsymbol{x}} \tilde{p}^*(\boldsymbol{x}) d\boldsymbol{x}$. This problem is ubiquitous in machine leaning. For example, in Bayesian Inference, $\tilde{p}^*(\boldsymbol{x})$ corresponds to the product of prior and likelihood.

As we can not sample directly from distribution $p^*(\boldsymbol{x})$ or use it for inference, a common approach is to use Variational Inference (VI) to approximate the target distribution $p^*(\boldsymbol{x})$ with a tractable distribution p such as multi-variate Gaussians (Blei et al., 2017; Regier et al., 2017) or Gaussian Mixture Models (GMM) (Miller et al., 2017; Guo et al., 2016; Zobay, 2014). The optimization problem to obtain this approximation is commonly framed as minimizing the reverse Kullback-Leibler divergence (KL)

$$\mathrm{KL}(p||p^*) = \int_{\boldsymbol{x}} p(\boldsymbol{x};\boldsymbol{\theta}) \log \left(\frac{p(\boldsymbol{x};\boldsymbol{\theta})}{p^*(\boldsymbol{x})} \right) d\boldsymbol{x} \qquad (1)$$

with respect to the parameters $\boldsymbol{\theta}$ of the approximation. This objective is typically evaluated on samples drawn from $p(\boldsymbol{x};\boldsymbol{\theta})$ and optimized by stochastic optimization algorithms (Fan et al., 2015; Gershman et al., 2012). However, in order to perform the KL minimization efficiently, $p(\boldsymbol{x})$ is often restricted to belong to a simple family of models or is assumed to have non-correlating degrees of freedom (Blei et al., 2017; Peterson & Hartman, 1989), which is known as the mean field approximation. Unfortunately, such restrictions can introduce significant approximation errors.

Our approach focuses on learning multivariate Gaussian Mixture Models (GMMs) with full covariance matrices for approximating the target distribution. GMMs are desirable for VI, because they are capable of representing any continuous probability density function arbitrarily well, while inference with GMMs is relatively cheap. Naturally, variational inference with GMM approximations has been considered in the past. However, in order to make the minimization of objective (1) feasible, previous work either assumed factorized (Jaakkola & Jordan, 1998; Bishop et al., 1998) or isotropic (Gershman et al., 2012) mixture components or applied boosting by successively adding and optimizing new components while keeping previously added components fixed (Miller et al., 2017; Guo et al., 2016).

As areas of high density $\tilde{p}^*(\boldsymbol{x})$ are initially unknown, Variational Inference can essentially be seen as a search problem that is inflicted by the exploration-exploitation dilemma which is typical for reinforcement learning or policy search problems. The algorithms need to explore the sample space in order to ensure that all relevant areas are covered while they also need to exploit the current approximation $p(\boldsymbol{x};\boldsymbol{\theta})$ in order to fine tune $p(\boldsymbol{x};\boldsymbol{\theta})$ in areas of high density. This exploration-exploitation based view is so far under-

[1]Computational Learning for Autonomous Systems, TU Darmstadt, Darmstadt, Germany [2]Machine Learning Lab, University of Lincoln, Lincoln, UK [3]Lincoln Center for Autonomous Systems, University of Lincoln, Lincoln, UK. Correspondence to: Oleg Arenz <oleg@robot-learning.de>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

developed in the Variational Inference community but essential to achieve good approximations with a small number of function evaluations.

Our method transfers information-geometric insights used in Policy Search (Deisenroth et al., 2013) and Reinforcement Learning (Sutton & Barto, 1998) on solving such exploration-exploitation dilemma efficiently. We therefore call our algorithm Variational Inference by Policy Search (VIPS). We extend the stochastic search method MORE (Abdolmaleki et al., 2016) to the variational inference setup and show that this version of MORE can efficiently learn single multivariate Gaussian variational distributions. We further extend the algorithm for training GMM distributions using a variational lower bound that enables us to train the mixture components and the coefficients individually. Our optimization starts from an initial mixture model (e.g. a Gaussian prior) and the number of components is adapted online by adding components in promising regions and by deleting components with negligible weight.

We compare our method to other state-of-the-art VI methods (Miller et al., 2017; Gershman et al., 2012) and show that our algorithm can find approximations of much higher quality with significantly less evaluations of $\tilde{p}^*(\boldsymbol{x})$. We further compare to existing sampling methods (Murray et al., 2010; Neal, 2003; Calderhead, 2014; Liu & Wang, 2016) and show that we can achieve similar sample quality with order of magnitudes less function evaluations.

2. Preliminaries

We will now formalize the problem statement and introduce relevant concepts from information-geometric policy search.

2.1. Problem formulation

As stated above, we want to minimize the KL divergence between the approximation p and the target distribution p^*. This direct minimization is infeasible as the normalization constant of p^* is unknown, however, it can be easily shown that the objective can be rewritten as

$$\mathrm{KL}(p||p^*) = \underbrace{\int_{\boldsymbol{x}} p(\boldsymbol{x};\boldsymbol{\theta}) \log \frac{p(\boldsymbol{x};\boldsymbol{\theta})}{\tilde{p}^*(\boldsymbol{x})} d\boldsymbol{x}}_{-\mathrm{ELBO}} + \log Z, \qquad (2)$$

where the term $\log Z$ can be ignored as it does not depend on $\boldsymbol{\theta}$. This objective is known as the negative value of the evidence lower bound (ELBO) used in many variational inference methods (Blei et al., 2017). As we want to use insights from policy search, we rewrite the ELBO as *reward maximization* problem with an additional entropy objective,

$$L(\boldsymbol{\theta}) = \int_{\boldsymbol{x}} p(\boldsymbol{x};\boldsymbol{\theta}) R(\boldsymbol{x}) d\boldsymbol{x} + H(p), \qquad (3)$$

where the reward is given by $R(\boldsymbol{x}) = \log \tilde{p}^*(\boldsymbol{x})$ and $H(p) = -\int_{\boldsymbol{x}} p(\boldsymbol{x};\boldsymbol{\theta}) \log p(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}$ is the entropy of p. Please note the swap in the sign due to the change from a minimization to a maximization problem. Hence, policy search algorithms can directly be applied to VI if they can incorporate an additional entropy objective. The ELBO objective can typically not be evaluated in closed form but is estimated by samples drawn from $p(\boldsymbol{x};\boldsymbol{\theta})$. Stochastic optimization can be used to optimize this sample-based objective (Blei et al., 2017).

2.2. Information Geometric Distribution Updates

Policy search algorithms must solve the exploration-exploitation dilemma when updating the policy, i.e., they must control how much information from the current sample set is integrated in the policy versus how much we rely on keeping the current exploration strategy to generate more information. Information-geometric trust regions can be used to effectively control this exploration-exploitation trade-off (Peters et al., 2010; Schulman et al., 2015; Abdolmaleki et al., 2017; 2016). We will heavily rely on a recent stochastic search algorithm called MORE (Abdolmaleki et al., 2016), that for the first time allows for a closed form solution of information geometric trust regions for Gaussian distributions. Stochastic search is a special case of policy search where the policy can be interpreted as search distribution $p(\boldsymbol{x})$ in the parameter space $\boldsymbol{x}$ of a low-level policy.

The MORE algorithm solves a constraint optimization problem where the objective is given by maximizing the average reward. The information-geometric trust region is implemented by limiting the KL-divergence between the old and new search distribution which is equivalent to limiting the information gain of the policy update. Moreover, a second constraint limits the entropy loss of the distribution update to avoid a collapse of the search distribution. The resulting optimization program has the following form:

$$\begin{aligned} &\underset{p(\boldsymbol{x})}{\text{maximize}} \int_{\mathbf{x}} p(\boldsymbol{x}) R(\boldsymbol{x}) d\boldsymbol{x}, \\ &\text{s. t.} \quad \mathrm{KL}\big(p||q\big) \leq \epsilon, \quad H\big(q\big) - H\big(p\big) \leq \gamma. \end{aligned} \qquad (4)$$

Here, $q(\boldsymbol{x})$ is the old search distribution, $R(\boldsymbol{x})$ is the reward and H the entropy of the distribution. The optimization problem can be solved using Lagrangian multipliers. The solution for $p(\boldsymbol{x})$ is given by

$$p(\boldsymbol{x}) \propto q(\boldsymbol{x})^{\frac{\eta}{\eta+\omega}} \exp\left(R(\boldsymbol{x})\right)^{\frac{1}{\eta+\omega}}, \qquad (5)$$

where η and ω are Lagrangian multipliers, which can be found by solving the convex dual optimization problem (Abdolmaleki et al., 2016).

2.3. Fitting a Reward Surrogate

In order to use Equation 5, Gaussianity needs to be enforced. This can be either performed by fitting a Gaussian (Daniel et al., 2012; Kupcsik et al., 2017) on samples that are weighted by Equation 5, which is prone to overfitting (as we need to fit a full covariance matrix) or by fitting a surrogate $\tilde{R}(\boldsymbol{x}) \approx R(\boldsymbol{x})$ that is compatible to the Gaussian distribution (Abdolmaleki et al., 2016). As the Gaussian distribution is log linear in quadratic features of $\boldsymbol{x}$, the surrogate needs to be a quadratic approximation of $R(\boldsymbol{x})$, i.e.,

$$\tilde{R}(\boldsymbol{x}) = -0.5\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{a}^T\boldsymbol{x} + a_0.$$

The parameters $\boldsymbol{A}$, $\boldsymbol{a}$ and a_0 are learned from the *current sample set* using linear regression. The surrogate therefore only needs to be a *local approximation* of the reward function. Using $\tilde{R}(\boldsymbol{x})$ for $R(\boldsymbol{x})$ in Equation 5 yields a Gaussian distribution for $p(\boldsymbol{x})$ where the mean $\boldsymbol{\mu}$ and the *full covariance matrix* $\boldsymbol{\Sigma}$ can be evaluated in closed form. For the exact equations we refer to Abdolmaleki et al. (2016).

3. Variational Inference by Policy Search

As VI uses samples to evaluate the ELBO, VI is essentially a search problem where we need to find samples in areas of high density of p^* but also make sure that these samples are distributed with the correct entropy. Interpreting VI as search problem, the current approximation $p(\boldsymbol{x};\boldsymbol{\theta})$ is used to search in the space of $\boldsymbol{x}$. We can obtain samples from our current search distribution $p(\boldsymbol{x};\boldsymbol{\theta})$ and use them to update $p(\boldsymbol{x};\boldsymbol{\theta})$ such that it becomes a better approximation of $p^*(\boldsymbol{x})$. Hence, VI is inflicted by the exploration-exploitation dilemma in a similar way as reinforcement learning and policy search algorithms. In this paper, we want to use information-geometric trust regions for controlling the exploration-exploitation trade-off in the VI objective.

Information-geometric trust regions have been shown to yield efficient closed form updates for Gaussian distributions (Abdolmaleki et al., 2016). However, in order to cope with more complex, multimodal distributions, we will extend the information-geometric updates to Gaussian Mixture Models. Hence, our variational distribution is given by a GMM

$$p(\boldsymbol{x}) = \sum_o p(o)p(\boldsymbol{x}|o),$$

where o is the index of the mixture component, $p(o)$ are the mixture weights and $p(\boldsymbol{x}|o) = \mathcal{N}(\boldsymbol{\mu}_o, \boldsymbol{\Sigma}_o)$ is a multivariate normal distribution with mean $\boldsymbol{\mu}_o$ and *full covariance* matrix $\boldsymbol{\Sigma}_o$. To improve readability, we will omit the parameter vector $\boldsymbol{\theta}$ when writing the distribution $p(\boldsymbol{x})$ in most cases. However, as we are dealing with mixture models, it should be noted that $\boldsymbol{\theta}$ consists of the mean vectors, covariance matrices and mixture weights of all components.

We will first introduce our objective including the trust regions and subsequently introduce a variational lower bound to decompose the objective into tractable optimization problems for the individual mixture components and the mixture coefficients. The number of components is automatically adapted by deleting components that have low weight and by creating new components in promising regions.

3.1. Objective

We approximate the target distribution $p^*(\boldsymbol{x})$ by minimizing $L(\boldsymbol{\theta})$ given in Equation 3 on the current set of samples. As we will use local approximations of the reward function around each component, we introduce individual trust regions for each component and for the coefficients. The resulting optimization problem has the form

$$\begin{aligned} \underset{p(\boldsymbol{x}|o),p(o)}{\text{maximize}} \quad & \int_{\mathbf{x}} p(\boldsymbol{x})\big(R(\boldsymbol{x}) - \log p(\boldsymbol{x})\big)d\boldsymbol{x}, \\ \text{subject to} \quad & \forall_o : \int_{\mathbf{x}} p(\boldsymbol{x}|o) \log \frac{p(\boldsymbol{x}|o)}{q(\boldsymbol{x}|o)} \leq \epsilon(o), \\ & \sum_o p(o) \log \frac{p(o)}{q(o)} \leq \epsilon_w, \end{aligned}$$

where $q(o)$ and $q(\boldsymbol{x}|o)$ are the old mixture weights and components, respectively, and ϵ_w and $\epsilon(o)$ upper-bound the corresponding KL-divergences. However, the occurrence of the log-density of the GMM, $\log p(\boldsymbol{x})$, prevents us from updating each component independently.

3.2. Variational Lower Bound

By introducing an auxiliary distribution $\tilde{p}(o|\boldsymbol{x})$, the objective of the optimization problem can be decomposed into a lower bound $U(\boldsymbol{\theta}, \tilde{p})$ and an expected KL-term, namely

$$J(\boldsymbol{\theta}) = U(\boldsymbol{\theta}, \tilde{p}) + \mathbb{E}_{p(\boldsymbol{x})}\Big[\mathrm{KL}\big(p(o|\boldsymbol{x})||\tilde{p}(o|\boldsymbol{x})\big)\Big], \quad (6)$$

with

$$\begin{aligned} U(\boldsymbol{\theta}, \tilde{p}) = \int_{\mathbf{x}} \sum_o p(\boldsymbol{x}, o)\big(R(\boldsymbol{x}) - \log p(\boldsymbol{x}, o) \\ + \log \tilde{p}(o|\boldsymbol{x})\big)d\boldsymbol{x} \end{aligned}$$

and

$$\mathrm{KL}\big(p(o|\boldsymbol{x})||\tilde{p}(o|\boldsymbol{x})\big) = \sum_o p(o|\boldsymbol{x}) \log \frac{p(o|\boldsymbol{x})}{\tilde{p}(o|\boldsymbol{x})}.$$

We also used the identity $p(\boldsymbol{x}, o) = p(\boldsymbol{x}|o)p(o)$ to keep the notation uncluttered. Eq. 6 can be easily verified by using Bayes theorem, i.e., by setting $\log p(o|\boldsymbol{x}) = \log p(\boldsymbol{x}, o) - \log p(\boldsymbol{x})$, all introduced terms will vanish and only the original objective remains, see supplement. The second term in Eq. 6 is always greater or equal zero. Hence,

$U(\boldsymbol{\theta}, \tilde{p})$ is a lower bound on the objective. This decomposition is closely related to the decomposition used in the expectation maximization (EM) algorithm (Bishop, 2006). However, as we want to minimize the reverse KL instead of the maximum likelihood (which corresponds to the forward KL) that is the objective in standard EM, the KL used for $\tilde{p}(o|\boldsymbol{x})$ also needs to be reversed to obtain the lower bound.

Following the standard EM procedure, we can maximize the objective function by iteratively maximizing the lower bound (M-step) and tightening the lower bound (E-step). The E-step can be computed by setting $\tilde{p}(o|\boldsymbol{x}) = p(\boldsymbol{x}|o)p(o)/p(\boldsymbol{x})$ using the current GMM. Hence, after the E-step, the lower bound is tight as the KL is set to 0. Consequently, improving the lower bound also improves the original objective $J(\boldsymbol{\theta})$ at each EM iteration. It can be easily seen that the lower bound does not contain the term $\log p(\boldsymbol{x})$ anymore and decomposes into individual terms for the weight distribution and the single components such that the updates can be performed independently. Moreover, an interesting observation is the term $\log \tilde{p}(o|\boldsymbol{x})$ acts as additional reward. After each sampling process from $p(\boldsymbol{x})$, we perform 10 EM-iterations to fine tune the mixture components with the newly obtained samples.

3.3. M-step for Component Updates

When updating a single component $p(\boldsymbol{x}|o)$, maximizing U is equivalent to maximizing

$$U_o(\boldsymbol{\mu_o}, \boldsymbol{\Sigma_o}) = \int_{\mathbf{x}} p(\boldsymbol{x}|o)\Big(R(\boldsymbol{x}) + \log \tilde{p}(o|\boldsymbol{x})\Big)d\boldsymbol{x} + H\big(p(\boldsymbol{x}|o)\big) + \eta(o)\left(\epsilon(o) - \int_{\mathbf{x}} p(\boldsymbol{x}|o)\log\frac{p(\boldsymbol{x}|o)}{q(\boldsymbol{x}|o)}d\boldsymbol{x}\right),$$

where we already added the Lagrangian multiplier $\eta(o)$ of the KL constraint for component o. This optimization problem is very similar to the optimization problem that is solved in MORE (Eq.4), which becomes evident when defining

$$r_o(\mathbf{x}) = R(\boldsymbol{x}) + \log \tilde{p}(o|\boldsymbol{x}) \tag{7}$$

as reward function. In fact, the only difference compared to MORE is that the entropy of the component enters the optimization via a constant factor 1 rather than a Lagrangian multiplier. Hence, we only need to optimize the Lagrangian multiplier of the KL constraint, $\eta(o)$ as ω is set to 1 in the MORE equations, see Eq. 5.

We refer to the supplement and to the original MORE paper (Abdolmaleki et al., 2016) for the closed form updates of this optimization problem based on quadratic reward surrogates $\tilde{r}_o(\boldsymbol{x}) \approx r_o(\boldsymbol{x})$. Note that in difference to the original MORE paper, we use a standard linear regression to obtain the quadratic models. The used dimensionality reduction technique reported by Abdolmaleki et al. (2016) was not needed to achieve satisfactory results.

3.4. M-step for Weight Updates

Maximizing U with respect to $p(o)$ is equivalent to maximizing

$$U_w\left(p(o)\right) = \sum_o p(o) r_w(o) + H\big(p(o)\big) + \eta_w\left(\epsilon_w - \sum_o p(o)\log\frac{p(o)}{q(o)}\right),$$

where we already added the Lagrangian multiplier for the KL constraint on the weights and

$$r_w(o) = \int_x p(\boldsymbol{x}|o) r_o(\mathbf{x}) d\boldsymbol{x} + H(p(\mathbf{x}|o)).$$

This optimization problem corresponds to optimizing a discrete distribution with an additional entropy objective. The solution for $p(o)$ is obtained by

$$p(o) \propto q(o)^{\frac{\eta_w}{\eta_w+1}} \exp\left(r_w(o)\right)^{\frac{1}{\eta_w+1}}. \tag{8}$$

Please refer to the supplement for the dual function and gradient of this optimization problem.

3.5. Sample Reuse

Samples are used for approximating $r_o(\boldsymbol{x})$ for the component updates as well as $r_w(o)$ for the weight updates.

We fit a quadratic surrogate to approximate $r_o(\boldsymbol{x})$ using linear regression, where the samples relate to the independent variables. The surrogate should be most accurate in the vicinity of $p(\boldsymbol{x}|o)$ which can be achieved by using independent variables that are distributed according to $p(\boldsymbol{x}|o)$. However, we also want to make use of samples from previous iterations or different components. We therefore perform weighted least squares, where the weight of sample i is given by the self-normalizing importance weight

$$w_i(o) = \frac{1}{Z}\frac{p(\boldsymbol{x}_i|o)}{z(\boldsymbol{x}_i)}, \qquad Z = \sum_i \frac{p(\boldsymbol{x}_i|o)}{z(\boldsymbol{x}_i)}.$$

The sampling distribution $z(\boldsymbol{x})$ for the current set of N_s samples is a Gaussian Mixture Model given by

$$z(\boldsymbol{x}) = \sum_{i=1}^{N_s} \frac{1}{N_s}\mathcal{N}_i(\boldsymbol{x}),$$

where $\mathcal{N}_i(\boldsymbol{x})$ corresponds to the normal distribution that was used to obtain sample i. For better efficiency, we include several samples from each sampling component $\mathcal{N}_i(\boldsymbol{x})$ such that $z(\boldsymbol{x})$ comprises less than N_s different component.

We approximate $r_w(o)$ for the weight updates by using an importance-weighted Monte Carlo estimate based on the same importance weights that are used for the component updates. Hence, we replace $r_w(o)$ in Eq. 8 by

$$\tilde{r}_w(o) = \sum_{i=1}^{N_s} w_i(o) r_o(\mathbf{x}_i) + H(p(\mathbf{x}|o)).$$

The set of active samples is replaced directly after new samples have been drawn and is held constant during the EM iterations. For all our experiments we selected the $3 \cdot N_{new}$ most recent samples, where N_{new} corresponds to the number of samples that have been drawn during the last round of sampling. Each round of sampling consists of drawing $10 \cdot D$ samples from each mixture component, where D corresponds to the number of dimensions of $\boldsymbol{x}$.

3.6. Adding and Deleting Mixture Components

In order to adapt the complexity of the GMM to the complexity of the target distribution, we initialize our algorithm with only one mixture component with high variance and gradually increase the number of mixture components. We consider the locations of all previous samples as candidates for new components. Every n_{add} iterations, we heuristically choose the most promising candidate and use its location as mean for a newly added component. The covariance matrix is initialized by interpolating the covariance matrices of neighboring components using the responsibilities.

As we want the new component to eventually achieve high weight, we want to add it at an area where its component-specific reward $r_o(\boldsymbol{x})$ will become large. We therefore try to find an area that has high likelihood under the target distribution but also yields high log-responsibilities for the newly added component, see Equation 7. However, the responsibilities of a newly initialized component hardly relate to the responsibilities it will eventually achieve. Instead, we choose the candidate that maximizes the score $e_i = \log \tilde{p}^*(\boldsymbol{x}_i) - \max(\log p(\boldsymbol{x}_i), \max_j \log p(\mathbf{x}_j) - \gamma)$. The second term prefers locations that are little covered by the current approximation and behaves similar to the log-responsibilities which also saturate for far away areas. Without such saturation, the second term might dominate the score when the target distribution has heavy tails.

In order to add components without impairing the stability of the optimization, we initialize them with very low weight such that their effect on the approximation is negligible. As we draw the same number of samples from each component irrespective of their weight, such low weight components can still improve and eventually contribute to the approximation. However, keeping unnecessary components is costly in terms of computational cost and sample efficiency. We therefore delete components that did not have mentionable effect on the approximation during the last n_{del} iterations.

The full algorithm is outlined in Algorithm 1. An open-source implementation is available online[1].

Algorithm 1 Variational Inference by Policy Search

Input: initial parameters $\boldsymbol{\theta}_0$
for $j = 1$ **to** $maxIter$ **do**
  Add and delete components according to heuristics. Store new parameters in $\boldsymbol{\theta}_j$
  Draw samples $\boldsymbol{s}_i$ from each component and store them along with $r_i = \log \tilde{p}^*(\boldsymbol{s}_i)$ and the parameters of the responsible component.
  $(\boldsymbol{s}^\subset, r^\subset, z^\subset(\boldsymbol{x})) \leftarrow$ select_active_samples()
  compute $\boldsymbol{z}_j = z^\subset(\boldsymbol{s}^\subset)$
  for $k = 1$ **to** $maxIterEM$ **do**
    recompute $\boldsymbol{p} = p(\boldsymbol{s}^\subset; \boldsymbol{\theta}_j), \tilde{\boldsymbol{p}} = p(o|\boldsymbol{s}^\subset; \boldsymbol{\theta}_j)$
    $\boldsymbol{\theta}_j \leftarrow$ update_weights$(\boldsymbol{p}, \boldsymbol{z}_j, \tilde{\boldsymbol{p}}, \boldsymbol{s}^\subset, r^\subset, \boldsymbol{\theta}_j)$
    recompute $\boldsymbol{p} = p(\boldsymbol{s}^\subset; \boldsymbol{\theta}_j), \tilde{\boldsymbol{p}} = p(o|\boldsymbol{s}^\subset; \boldsymbol{\theta}_j)$
    for each component **do**
      $\boldsymbol{\theta}_j \leftarrow$ update_component$(\boldsymbol{p}, \boldsymbol{z}_j, \tilde{\boldsymbol{p}}, \boldsymbol{s}^\subset, r^\subset, \boldsymbol{\theta}_j)$
    end for
  end for
end for

4. Related Work

Although MCMC samplers can not directly be used for approximating distributions, they are for many applications the main alternative to VI. Prominent examples of gradient-free MCMC methods include (random walk) Metropolis Hasting (Hastings, 1970), Gibbs sampling (Geman & Geman, 1984), slice sampling (Neal, 2003) and elliptical slice sampling (Murray et al., 2010; Nishihara et al., 2014). If the gradient of the target distribution is available, Hamiltonian MCMC (Duane et al., 1987) and the Metropolis-adjusted Langevin algorithm (Roberts & Stramer, 2002) are also popular choices. The No-U-Turn sampler (NUTS) (Hoffman & Gelman, 2014) is a notable variant of Hamiltonian MCMC that is appealing for not requiring hyper-parameter tuning. While many of these MCMC methods have problems with multimodal distributions in terms of mixing time, other methods use multiple chains and can therefore better explore multimodal sample spaces (Neal, 1996; Nishihara et al., 2014; Calderhead, 2014).

Many VI methods are only applicable to Gaussian variational distributions (Fan et al., 2015; Regier et al., 2017). The approach by Fan et al. (2015) can learn Gaussians with full covariance matrices using fast second order optimization. This idea has been extended by Regier et al. (2017) to trust region optimization. However, in difference to our approach, an euclidean trust region is used in parameter space

[1] https://github.com/OlegArenz/VIPS

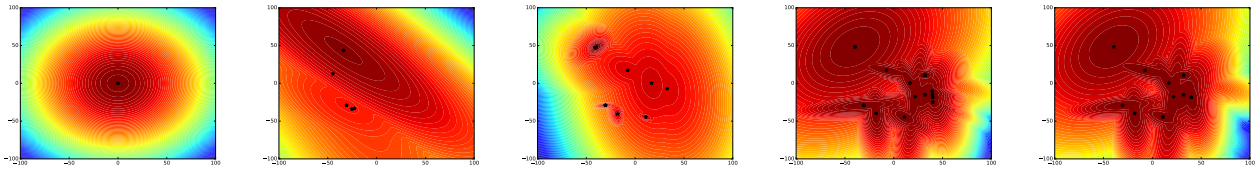


Figure 1: We start with an initial isotropic Gaussian distribution (left) and iteratively improve the approximation and add additional components in order to approximate the desired distribution (right). The three plots in the middle visualize the approximation directly after adding the fifth, tenth and twentieth component.

of the variational distribution. Such approach requires the computation of the Hessian of the objective which is only tractable for mean-field approximations of single Gaussian distributions. In contrast, we use the trust regions directly on the change of the distributions instead of the change of the parameters of the distribution. The information geometric trust regions in this paper allow for efficient estimation of GMMs with full covariance matrices without requiring gradient information from p^*.

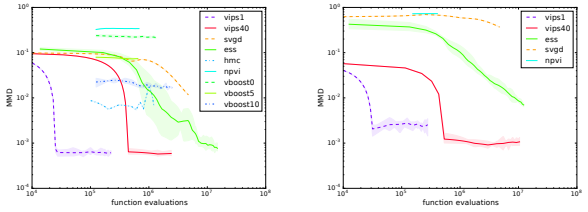
Black-box Variational Inference methods do not make strong assumptions on the model family (Salimans & Knowles, 2013; Ranganath et al., 2014) and can therefore also be used for learning GMM approximations. Salimans & Knowles (2013) derive a fixed point update of the natural parameters of a distribution from the exponential family that corresponds to a Monte-Carlo estimate of the gradient of Eq. (1) preconditioned by the inverse of their empirical covariance. By making structural assumptions on the target distribution, they extend their method to mixture models and show its applicability to bivariate GMMs. Ranganath et al. (2014) also use Monte-Carlo gradient estimates of Eq. (1) and apply Rao-Blackwellization and control variates to reduce its variance. The work of Weber et al. (2015) already explored the use of Reinforcement Learning for VI but formalizing VI as sequential decision problem. However, only simple policy gradient methods have been proposed in this context which are unsuitable for learning GMMs.

Closely related to our work are two recent approaches for Variational Inference that concurrently explored the idea of applying boosting to make the training of GMM approximations tractable (Miller et al., 2017; Guo et al., 2016). These methods start by minimizing the ELBO objective for a single component and then successively add and optimize new components and learn an optimal weighting between the previous mixture and the newly added component. However, they do not use information-geometric trust region to efficiently explore the sample space and therefore have problems finding all the modes as well as accurate estimates of the covariance matrices. Non-parametric variational inference (NPVI) (Gershman et al., 2012) learns GMMs with uniform weights using a second-order approximation of the ELBO for efficient gradient updates. However, this method only allows for mean field approximations for the mixture components which is a severe limitation as shown in our comparisons. GMMs are also used by Zobay (2014) where an approximation of the GMM entropy is used to make the optimization tractable. The optimization is gradient-based and does not consider exploration of the sample space. It is therefore limited to rather low dimensional problems.

5. Experiments

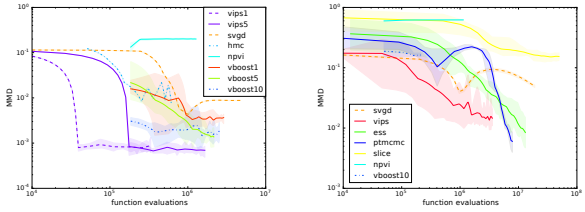
We evaluate VIPS with respect to efficiency of the optimization as well as quality of the learned approximations. For assessing efficiency, we focus on the number of function evaluations, but also include a comparison with respect to the wall-clock time. As the ELBO objective is hard to use for comparisons as it depends on the current sample set, we assess the quality of the approximation by comparing samples drawn from the learned model with ground-truth samples based on their Maximum Mean Discrepancy (MMD) (Gretton et al., 2012). Please refer to the supplement how the MMD and the ground-truth samples are computed. Please note that the computation of the ground-truth samples is based on generalized elliptical slice sampling (GESS) (Nishihara et al., 2014) which is in most cases computationally very expensive. Due to the huge computational demands, we do not consider GESS as competitor.

We compare our method to a variety of state-of-the-art MCMC and VI approaches on unimodal and multimodal problems, namely we compare to Variational Boosting (Miller et al., 2017), Non-Parametric Variational Inference (Gershman et al., 2012), Stein Variational Gradient Descent (Liu & Wang, 2016), Hamiltonian Monte Carlo (Duane et al., 1987), Slice Sampling (Neal, 2003), Elliptical Slice Sampling (Murray et al., 2010), Parallel Tempering MCMC (Calderhead, 2014) and—indirectly, since it is used for initializing variational boosting—black box variational inference (Salimans & Knowles, 2013). However, due to the high computational demands, we do not compare to every method on each experiment but rather select promising candidates based on the sampling problem or on the preliminary experiments that we had to conduct for hyper-parameter tun-



(a) German Credit (b) Breast Cancer

Figure 2: Comparison of VIPS to other VI and MCMC methods for two logistic regression tasks. VIPS considerably outperformed all other methods already when using only one mixture component (VIPS1) and could only be slightly improved by the GMM on the breast cancer domain.



(a) stop-and-frisk (b) 10-link planar robot

Figure 3: (a) On the *stop-and-frisk* dataset, VIPS1 closely approximates the posterior with roughly 33000 function evaluations. (b) On the 10-link robot experiment VIPS learns a significantly better approximation than competing VI methods and achieves similar sample quality to MCMC.

ing. For VIPS, we use the same hyper-parameters for all experiments. However, we do not add new components if it would increase the number of components above a certain threshold and evaluate different values for this threshold.

We perform three experiments on (essentially) unimodal sampling problems taken from related work. We perform Bayesian logistic regression on the *German Credit* and *Breast Cancer* datasets (Lichman, 2013) as described in (Nishihara et al., 2014) and approximate the posterior of the hierarchical Poisson GLM described in (Miller et al., 2017). We designed two challenging toy tasks for evaluating our approach on multimodal problems: sampling from an unknown twenty-dimensional GMM with full covariance matrices and distant modes, and sampling the joint configurations of a ten-dimensional planar robot.

We start the experimental evaluation of VIPS by visualizing the iterative improvements of the GMM approximation for the task of approximating an unknown two-dimensional GMM. Figure 1 shows the log-probability densities of the learned approximation during the optimization and the target distribution (right). We can see that VIPS gradually adds more components and improves the GMM. The final fit with 20 components closely approximates the target distribution.

5.1. Bayesian Logistic Regression

We perform two experiments for binary classification on the *German credit* and *breast cancer* datasets (Lichman, 2013). For the *German credit* dataset twenty-five parameters are learned whereas the *breast cancer* dataset is thirty-one dimensional. We standardize both datasets and perform linear logistic regression where we put zero-mean Gaussian priors with variance 100 on all parameters.

On the *German credit* dataset we compare against NPVI, ESS, SVGD, HMC and variational boosting. For variational boosting we examine rank-0, rank-5 and rank-10 approximations of the covariance matrices. We also performed experiments with full rank approximation but the optimization always failed after few iterations. For our approach we examine two variants, VIPS1 which learns only a single component and VIPS40 which stops adding new components after forty components. Figure 2a shows that the sample quality achieved by VIPS is unmatched by any variational inference method and ESS needs more than two orders of magnitude more function evaluation to achieve a similar MMD to VIPS1. However, we could not measure an advantage of using a GMM instead of single Gaussian distribution on this dataset. As VIPS1 is only learning a single Gaussian, it is also more data-efficient than VIPS40. This result also illustrates the importance of using Gaussian distributions with an accurate estimation of the full covariance as VIPS1 could already outperform competing methods.

Figure 2b depicts the achieved MMDs on the *breast cancer* dataset. We only compared to the most promising methods from the *German credit* dataset. Although the posterior distribution is unimodal, the GMM variant of our method learns a slightly better approximation than VIPS1, presumably by matching higher order moments of the posterior.

5.2. Multi-level Poisson GLM

We evaluate our method on the problem of learning the posterior of a hierarchical Poisson GLM on the 37-dimensional *stop-and-frisk* dataset, where we refer to (Miller et al., 2017) for the description of the hierarchical model. We compared VIPS1 and VIPS5 to HMC, SVGD, variational boosting and non-parametric variational inference. The MMD with respect to the baseline samples is shown in Figure 3a.

5.3. Planar Robot

We evaluate our approach on a planar robot with ten degrees of freedom. We want to sample joint configurations such that the end-effector of the robot is close to a desired position at $x = 7$ and $y = 0$. The likelihood of a configu-

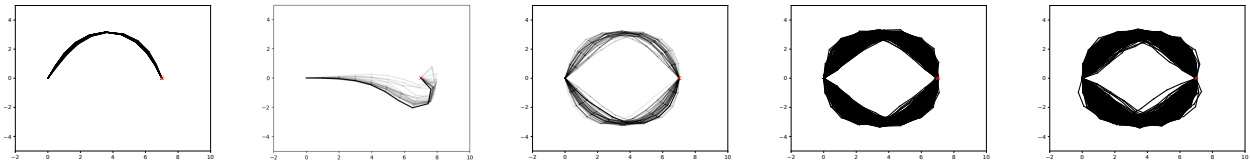
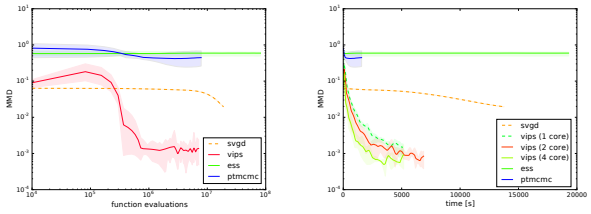


Figure 4: The first three plots visualize typical component means and weights learned by NPVI, VBOOST and VIPS. Components with higher weight are drawn darker. The two rightmost plots show the samples drawn from the VIPS approximation and ground-truth samples, that have been collected during two days on 120 cores using GESS.



(a) MMD plotted over iterations (b) MMD plotted over time

Figure 5: On the GMM experiment, VIPS was able to closely approximate the true mixture model and reliably discovers all ten components. When the number of computing cores is small compared to the components, VIPS can scale almost linearly with the number of cores.

ration is a Gaussian distribution in Cartesian end-effector space with a variance of $1\mathrm{e}{-4}$ in both dimensions. We assume a zero mean Gaussian prior in configuration space, where the first joint has variance 1 and the remaining joints have variance 0.04. We compare VIPS40 with ESS, parallel tempering MCMC, SVGD, slice sampling, NPVI and VBOOST. The MMD is shown in Figure 3b. Figure 4 visualizes the learned models of NPVI, VBOOST and VIPS and compares the ground-truth samples with the samples obtained by VIPS. All other VI methods can not represent the complex structure of the modes and get attracted by single modes resulting in bad approximations. We believe the reason for this behavior is a missing principled treatment of the exploration-exploitation trade-off. VIPS achieves a high sample quality that is comparable to MCMC methods in order of magnitude less function evaluations than MCMC.

5.4. Gaussian Mixture Model

We evaluate our method on the task of approximating unknown, randomly generated 20-dimensional GMMs comprising ten components. Each dimensions of the component means is drawn uniformly in the interval $[-50, 50]$. The covariance matrices are given by $\mathbf{\Sigma} = \mathbf{A}^\top \mathbf{A} + \mathbf{I}_{20}$ where each entry of the 20×20-dimensional matrix $\mathbf{A}$ is sampled from a normal distribution with mean 0 and variance 20. Note that each component of the target distribution can have a highly correlated covariance matrix, which is even a problem for the tested MCMC methods. Figure 5 shows the MMD of VIPS40, SVGD, ESS and parallel tempering MCMC plotted over the number of iterations as well as wall-clock time. VIPS was the only method that could reliable be applied to this task. We also briefly evaluated the scaling of the performance with the number of cores. Although parallel computing is not the focus of this paper, the possibility of performing independent updates of the mixture components suggests that the method can make good use of multi-threading. Figure 5b shows that VIPS scales almost linearly with the number of cores at least if their number is small, showing the potential of parallelizing our algorithm.

6. Conclusion

VIPS is motivated by the insight from stochastic search that information-geometric trust regions allow for controlled exploration and stable optimization. We transfered the stochastic search method MORE to the field of variational inference and demonstrated that it is significantly more efficient than state-of-the-art approaches of VI for learning mean and full covariance of a multi-variate normal distribution. Based on our variant of MORE, we derived a novel method for learning GMM approximations and demonstrated that it is capable of learning high quality approximations of complex, multimodal distributions with a limited amount of function evaluations. Our method makes little assumptions on the unnormalized density function of the desired distribution and is thereby applicable to non-differentiable problems.

However, for higher-dimensional problems (e.g. more than 100 dimensions) fitting quadratic surrogates may require too many samples which could be alleviated by using gradient information for constraining the surrogate. Furthermore, learning diagonal surrogates can be preferable for such problems due to better sample- and computational efficiency.
For exploration, we start with an approximation with high entropy and decrease it slowly during optimization which can miss modes that are not discovered in the beginning. Actively sampling unexplored regions would result in an anytime algorithm, capable of further improving the approximation in the later stages of optimization.

Acknowledgements

This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 645582 (RoMaNS). Calculations for this research were conducted on the Lichtenberg high performance computer of the TU Darmstadt.

References

Abdolmaleki, A., Lioutikov, R., Lua, N., Reis, L. P., Peters, J., and Neumann, G. Model-Based Relative Entropy Stochastic Search. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 153–154, 2016.

Abdolmaleki, A., Price, B., Lau, N., Reis, L. P., and Neumann, G. Deriving and Improving CMA-ES with Information Geometric Trust Regions. In *The Genetic and Evolutionary Computation Conference (GECCO 2017)*, July 2017.

Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, 2006.

Bishop, C. M., Lawrence, N. D., Jaakkola, T., and Jordan, M. I. Approximating Posterior Distributions in Belief Networks Using Mixtures. In *Advances in Neural Information Processing Systems*, pp. 416–422, 1998.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 2017.

Calderhead, B. A General Construction for Parallelizing Metropolis-Hastings Algorithms. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, Nov 2014.

Daniel, C., Neumann, G., and Peters, J. Hierarchical Relative Entropy Policy Search. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

Deisenroth, M. P., Neumann, G., and Peters, J. A Survey on Policy Search for Robotics. *Foundations and Trends in Robotics*, pp. 388–403, 2013.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987.

Fan, K., Wang, Z., Beck, J., Kwok, J. T., and Heller, K. Fast Second-order Stochastic Backpropagation for Variational Inference. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NIPS'15, pp. 1387–1395, 2015.

Geman, S. and Geman, D. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):721–741, 1984.

Gershman, S. J., Hoffman, M. D., and Blei, D. M. Nonparametric Variational Inference. In *Proceedings of the 29th International Conference on Machine Learning*, 235–242, 2012.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A Kernel Two-sample Test. *Journal of Machine Learning Research*, 13:723–773, March 2012. ISSN 1532-4435.

Guo, F., Wang, X., Fan, K., Broderick, T., and Dunson, D. B. Boosting Variational Inference. *arXiv:1611.05559v2 [stat.ML]*, 2016.

Hastings, W. K. Monte Carlo Sampling Methods using Markov Chains and their Applications. *Biometrika*, 57 (1):97–109, 1970.

Hoffman, M. D. and Gelman, A. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1): 1593–1623, 2014.

Jaakkola, T. S. and Jordan, M. I. Improving the Mean Field Approximation via the Use of Mixture Distributions. *Learning in Graphical Models*, 89:163–174, 1998.

Kupcsik, A., Deisenroth, M. P., Peters, J., Loh, A. P., Vadakkepat, P., and Neumann, G. Model-Based Contextual Policy Search for Data-Efficient Generalization of Robot Skills. *Artificial Intelligence*, December 2017.

Lichman, M. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Liu, Q. and Wang, D. Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29*, pp. 2378–2386. Curran Associates, Inc., 2016.

Miller, A. C., Foti, N. J., D'Amour, A., and Adams, R. P. Variational Boosting: Iteratively Refining Posterior Approximations. In *Proceedings of the International Conference on Machine Learning*, 2017.

Murray, I., Adams, R., and MacKay, D. Elliptical Slice Sampling. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 541–548, 2010.

Neal, R. M. Sampling from Multimodal Distributions using Tempered Transitions. *Statistics and Computing*, 6(4): 353–366, Dec 1996.

Neal, R. M. Slice Sampling. *The Annals of Statistics*, 31(3): 705–767, 06 2003. doi: 10.1214/aos/1056562461.

Nishihara, R., Murray, I., and Adams, R. P. Parallel MCMC with Generalized Elliptical Slice Sampling. *Journal of Machine Learning Research*, 15(1):2087–2112, January 2014.

Peters, J., Muelling, K., and Altun, Y. Relative Entropy Policy Search. In *Proceedings of the 24th National Conference on Artificial Intelligence (AAAI)*, 2010.

Peterson, C. and Hartman, E. Explorations of the Mean Field Theory Learning Algorithm. *Neural Networks*, 2 (6):457–494, 1989.

Ranganath, R., Gerrish, S., and Blei, D. Black Box Variational Inference. *Artificial Intelligence and Statistics*, pp. 814–822, 2014.

Regier, J., Jordan, M. I., and McAuliffe, J. Fast Black-box Variational Inference through Stochastic Trust-Region Optimization. In *Advances in Neural Information Processing Systems 30*, pp. 2399–2408, 2017.

Roberts, G. O. and Stramer, O. Langevin Diffusions and Metropolis-Hastings Algorithms. *Methodology and Computing in Applied Probability*, 4(4):337–357, 2002.

Salimans, T. and Knowles, D. A. Fixed-Form Variational Posterior Approximation through Stochastic Linear Regression. *Bayesian Analysis*, 8(4):837–882, 2013.

Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., and Moritz, P. Trust Region Policy Optimization. In *Proceedings of the International Conference on Machine Learning*, 2015.

Sutton, R. and Barto, A. *Reinforcement Learning: An Introduction*. MIT Press, Boston, MA, 1998.

Weber, T., Heess, N., Eslami, A., Schulman, J., Wingate, D., and Silver, D. Reinforced Variational Inference. In *Advances in Neural Information Processing Systems (NIPS) Workshops*, 2015.

Zobay, O. Variational Bayesian Inference with Gaussian-Mixture Approximations. *Electronic Journal of Statistics*, 8(1):355–389, 2014. doi: 10.1214/14-EJS887.

A. Derivation of the Lower Bound

As stated in the paper, the ELBO objective can be decomposed into a lower bound and an expected KL term, i.e.,

$$J(\boldsymbol{\theta}) = \int_{\mathbf{x}} \sum_o p(\boldsymbol{x}, o)\big(R(\boldsymbol{x}) - \log p(\boldsymbol{x}, o) + \log \tilde{p}(o|\boldsymbol{x})\big) d\boldsymbol{x} + \int_{\mathbf{x}} p(\boldsymbol{x}) \sum_o p(o|\boldsymbol{x}) \log \frac{p(o|\boldsymbol{x})}{\tilde{p}(o|\boldsymbol{x})}. \tag{9}$$

We can verify that this decomposition is valid by using the identity $\log p(o|\boldsymbol{x}) = \log p(\boldsymbol{x}, o) - \log p(\boldsymbol{x})$, i.e.,

$$\begin{aligned} J(\boldsymbol{\theta}) &= \int_{\mathbf{x}} \sum_o p(\boldsymbol{x}, o)\big(R(\boldsymbol{x}) - \log p(\boldsymbol{x}, o) \\ &\quad + \log \tilde{p}(o|\boldsymbol{x})\big) d\boldsymbol{x} \\ &\quad + \int_{\mathbf{x}} \sum_o p(\boldsymbol{x}) p(o|\boldsymbol{x})\big(\log p(\boldsymbol{x}, o) - \log p(\boldsymbol{x}) \\ &\quad - \log \tilde{p}(o|\boldsymbol{x})\big) d\boldsymbol{x}. \\ &= \int_{\mathbf{x}} \sum_o p(\boldsymbol{x}, o)\big(R(\boldsymbol{x}) - \log p(\boldsymbol{x})\big) d\boldsymbol{x} \\ &= \int_{\mathbf{x}} p(\boldsymbol{x})\big(R(\boldsymbol{x}) - \log p(\boldsymbol{x})\big) d\boldsymbol{x}. \end{aligned} \tag{10}$$

We can see that Eq. 10 corresponds to the original definition of $L(\boldsymbol{\theta})$ in the paper.

B. Computation of the MMD

The Maximum Mean Discrepancy (Gretton et al., 2012) is a nonparametric divergence between mean embeddings in a Reproducible Kernel Hilbert Space. We approximate the MMD between two sample sets $\mathbf{X}$ and $\mathbf{Y}$ as

$$\text{MMD}(\mathbf{X}, \mathbf{Y}) = \frac{1}{m^2} \sum_{i,j}^{m} k(x_i, x_j) + \frac{1}{n^2} \sum_{i,j}^{n} k(y_i, y_j) - \frac{2}{mn} \sum_{i}^{m} \sum_{j}^{n} k(x_i, y_i).$$

We use a squared exponential kernel given by

$$k(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{\alpha}(\mathbf{x} - \mathbf{y})^\top \boldsymbol{\Sigma} (\mathbf{x} - \mathbf{y})\right),$$

where $\boldsymbol{\Sigma}$ is a diagonal matrix where each entry is set to the median of squared distances within the ground-truth set and the bandwidth α is chosen depending on the problem. When ground-truth samples are not available, we apply GESS (Nishihara et al., 2014) with large values for burn-in, thinning and chain lengths to produce baseline samples that are regarded as ground-truth. Note that obtaining these ground-truth samples is computationally very expensive, taking up to 2 days of computation time on 120 CPU cores. We estimate the MMD based on ten thousand ground-truth samples and two thousand samples from the given sampling method. For MCMC methods, we choose the two thousand most promising samples by applying a sufficient amount of burn-in and using the largest thinning that keeps at least two thousand samples in the set.

C. Component Optimization

As the Lagrangian of the optimization problem for the component update corresponds to the Lagrangian of MORE (Abdolmaleki et al., 2016) with $\omega = 1$, the solution has the form

$$p(\boldsymbol{x}|o) \propto q(\boldsymbol{x}|o)^{\frac{\eta}{\eta+1}} \exp\left(\tilde{r}_o(\boldsymbol{x})\right)^{\frac{1}{\eta+1}}, \tag{11}$$

where we substituted $\omega = 1$ in Equation 5.

When the quadratic reward surrogate is given as

$$\tilde{r}_o(\boldsymbol{x}) = -\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{R}\boldsymbol{x} + \boldsymbol{x}^\top \boldsymbol{r},$$

the parameters $\boldsymbol{R}$ and $\boldsymbol{r}$ (which are learned with weighted least squares) correspond to the natural parameters of a multivariate normal distribution

$$p_r(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_r = \boldsymbol{R}^{-1}\boldsymbol{r}, \boldsymbol{\Sigma}_r = \boldsymbol{R}^{-1}) \propto \exp\left(\tilde{r}_o(\boldsymbol{x})\right).$$

Hence, the log-densities of $p(\boldsymbol{x}|o)$ are given by a linear interpolation of the log-densities of $q(\boldsymbol{x}|o)$ and $p_r(\boldsymbol{x})$, i.e.

$$\log p(\boldsymbol{x}|o) = \frac{\eta}{\eta+1} \log q(\boldsymbol{x}|o) + \frac{1}{\eta+1} \log p_r(\boldsymbol{x}) + \text{const.}$$

The natural parameters of $p(\boldsymbol{x}|o)$ are therefore given by

$$\boldsymbol{P} = \frac{1}{\eta+1}\left(\eta\boldsymbol{Q} + \boldsymbol{R}\right), \qquad \boldsymbol{p} = \frac{1}{\eta+1}\left(\eta\boldsymbol{q} + \boldsymbol{r}\right),$$

where $\boldsymbol{Q} = \boldsymbol{\Sigma}_q^{-1}$ and $\boldsymbol{q} = \boldsymbol{\Sigma}_q^{-1}\boldsymbol{\mu}_q$ are the natural parameters of $q(\boldsymbol{x}|o)$.

As a function of the Lagrangian multiplier η,

$$p(\boldsymbol{x}|o, \eta) = \mathcal{N}\left(\boldsymbol{x}|\boldsymbol{\mu}_p = \boldsymbol{P}(\eta)^{-1}\boldsymbol{p}(\eta), \boldsymbol{\Sigma}_p = \boldsymbol{P}(\eta)^{-1}\right)$$

defines an e-geodesic, i.e. a straight line connecting $q(\boldsymbol{x}|o)$ and $p_r(\boldsymbol{x})$ in logarithmic scale. During optimization we want to find the largest *step-size* η such that $p(\boldsymbol{x}|o, \eta)$ stays within the trust region. As we are minimizing a scalar on a convex function, a simple line-search would be feasible. However, the dual objective

$$G_o(\eta) = \eta\epsilon(o) + \eta \log Z(\boldsymbol{Q}, \boldsymbol{q}) - (\eta+1) \log Z(\boldsymbol{P}, \boldsymbol{p}),$$

where $\log Z(\boldsymbol{X}, \boldsymbol{x}) = -\frac{1}{2}(\boldsymbol{x}^\top \boldsymbol{X}^{-1}\boldsymbol{x} + \log|2\pi\boldsymbol{X}^{-1}|)$ is the log partition function of a Gaussian with natural parameters $\boldsymbol{X}$ and $\boldsymbol{x}$, as well as the gradient

$$\frac{dG_o(\eta)}{d\eta} = \epsilon(o) - \mathrm{KL}(p(\boldsymbol{x}|o,\eta)||q(\boldsymbol{x}|o))$$

can be computed with little overhead and hence we use L-BFGS for dual descent.

D. Weight Optimization

The optimization of the distribution over weights is similar to the optimization of the components but we are optimizing over a discrete distribution rather than a multivariate normal. Similar to the component optimization, the optimal distribution has the form

$$p(o) \propto q(o)^{\frac{\eta_w}{\eta_w+1}} \exp\left(\tilde{r}_w(o)\right)^{\frac{1}{\eta_w+1}}, \qquad (12)$$

and corresponds to a log-linear interpolation between the last distribution $q(o)$ and a distribution $p_r(o) \propto \exp(\tilde{r}_w(o))$ that is specified by the reward function. The optimal step-size η_w can be found by minimizing the dual

$$G_w(\eta_w) = \eta_w \epsilon_w + (1+\eta_w)\log\sum_o p(o|\eta_w)$$

based on the gradient

$$\frac{dG_w(\eta_w)}{d\eta_w} = \epsilon - \mathrm{KL}(p(o|\eta)||q(o)).$$

E. Hyper-parameters

Table 1 lists the hyper-parameters as well as their values for the experiments. We will now briefly discuss some of these hyper-parameters.

E.1. KL bounds

The trust regions are necessary for the component updates in order to ensure that the components stay within regions where their local reward surrogate $\tilde{r}_o(\boldsymbol{x})$ remains valid. As the reward surrogate is updated in each EM iteration, we also update the reference distribution $q(\boldsymbol{x}|o)$ after each EM iteration. However, this may allow the component to enter regions that are insufficiently covered by samples after several EM iterations which would result in bad local surrogates. We therefore compute the KL bound based on the effective number of samples within the active set, namely the KL bound is given by

$$\epsilon(o) = \min(1\mathrm{e}{-3}, 1\mathrm{e}{-5} \cdot n_{\text{eff}}(o)),$$

where the effective sample size is computed based on the importance weights

$$n_{\text{eff}}(o) = \frac{\left(\sum_{i=1}^{N_s} w_i(o)\right)^2}{\sum_{i=1}^{N_s} w_i(o)^2}.$$

Table 1: A list of the hyper-parameters of VIPS as well their values used during the experiments.

DESCRIPTION	VALUE
MAXIMUM NUMBER OF COMPONENTS	$1, 5, 40$
NUMBER OF EM ITERATIONS	10
KL BOUND FOR WEIGHTS	$1\mathrm{e}{-2}$
MAXIMUM KL BOUND FOR COMPONENTS	$1\mathrm{e}{-3}$
KL BOUND FACTOR FOR COMPONENTS	$1\mathrm{e}{-5}$
NUMBER OF SAMPLES PER COMPONENT	$10 \cdot D$
NUMBER OF INITIAL SAMPLES	$20, 20000$
SAMPLE REUSE FACTOR	3
ADDING RATE FOR COMPONENTS	30
DELETION RATE FOR COMPONENTS	300
MINIMUM WEIGHT	$1\mathrm{e}{-7}$
INITIAL WEIGHT	$1\mathrm{e}{-7}$
γ FOR ADDING-HEURISTIC	500
ℓ_2-REGULARIZATION FOR WLS	$1\mathrm{e}{-10}$

As we ignore the weights for sampling during training, the KL bound for the weights is not critical and could even be dropped. However, for the experiments we chose a KL bound of $\epsilon_w = 1\mathrm{e}{-2}$, because it seems sensible to prevent large jumps in the log responsibilities.

E.2. Samples

As stated in the paper, we draw $10D$ samples *per component* and roughly reuse the samples from the last 3 most recent iterations. For the experiments, we drew 20000 additional samples from the initial mixture at the beginning of the optimization for better initial exploration. However, we lowered this value to 20 for VIPS1 which often already converged after 20000 iterations.

E.3. Adding and Deleting Components

We added a single new component every third sampling iteration and initialized its weight to $1\mathrm{e}{-7}$. For computing the score e_i for deciding where to add the component, we use $\gamma = 500$. This hyper-parameter is probably the least intuitive to be chosen. When γ is too small, new modes may only be discovered when we have sampled close to their peak. However, when γ is too large we might add components at irrelevant regions, especially when the target distribution has heavy tails. However, we found $\gamma = 500$ to produce good results among all our experiments.

We delete a component when its weight was below $1\mathrm{e}{-7}$ for the last 300 EM-Iterations (i.e. 30 sampling iterations). We do not want to keep components with lower weight, because their effect on the approximation would be marginal.

References

Abdolmaleki, A., Lioutikov, R., Lua, N., Reis, L. P., Peters, J., and Neumann, G. Model-Based Relative Entropy Stochastic Search. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 153–154, 2016

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A Kernel Two-sample Test. *Journal of Machine Learning Research*, 13:723–773, March 2012. ISSN 1532-4435

Nishihara, R., Murray, I., and Adams, R. P. Parallel MCMC with Generalized Elliptical Slice Sampling. *Journal of Machine Learning Research*, 15(1):2087–2112, January 2014