Abstract
We propose a black-box variational inference method to approximate intractable distributions with an increasingly rich approximating class. Our method, variational boosting, iteratively refines an existing variational approximation by solving a sequence of optimization problems, allowing a trade-off between computation time and accuracy. We expand the variational approximating class by incorporating additional covariance structure and by introducing new components to form a mixture. We apply variational boosting to synthetic and real statistical models, and show that the resulting posterior inferences compare favorably to existing variational algorithms.

1. Introduction
Variational inference (VI) is a family of methods to approximate an intractable target distribution (typically known only up to a constant) with a tractable surrogate distribution (Blei et al., 2016; Jordan et al., 1999; Wainwright & Jordan, 2008). VI procedures typically minimize the Kullback-Leibler (KL) divergence between the approximation and target distributions by maximizing a tractable lower bound on the marginal likelihood. The approximating family is often fixed, and typically excludes the neighborhood surrounding the target distribution, which prevents the approximation from becoming arbitrarily close to the true posterior. In the context of Bayesian inference, this mismatch between the variational family and the true posterior often manifests as underestimating the posterior variances of the model parameters and the inability to capture posterior correlations (Wainwright & Jordan, 2008).

An alternative approach to posterior inference uses Markov chain Monte Carlo (MCMC) methods that approximate a target distribution with samples drawn from a Markov chain constructed to admit the target distribution as the stationary distribution. MCMC enables a trade-off between computation and accuracy: drawing more samples makes the approximation closer to the target distribution. However, MCMC algorithms typically must be run iteratively and it can be difficult to assess convergence to the true target. Furthermore, correctly specifying MCMC moves can be more algorithmically restrictive than optimization-based approaches.

To alleviate the mismatch between tractable variational approximations and complicated posterior distributions, we propose a variational inference method that iteratively allows the approximating family of distributions to become more complex. Under certain conditions, the proposed approximations are eventually expressive enough to represent the true target arbitrarily well (though we do not prove our algorithm attains such a universal approximation here). Thus, the practitioner can trade time fitting a posterior approximation for increased accuracy of posterior estimates. Our algorithm grows the complexity of the approximating class in two ways: 1) incorporating rich covariance structure, and 2) sequentially adding new components to the approximating distribution. Our method builds on black-box variational inference methods using the re-parameterization trick by adapting it to be used with mixture distributions. This allows our method to be applied to a variety of target distributions including those arising from non-conjugate model specifications (Kingma & Welling, 2014; Ranganath et al., 2014; Salimans & Knowles, 2013). We demonstrate empirically that our algorithm improves posterior estimates over other variational methods for several practical Bayesian models.

2. Variational Inference
Given a target distribution with density \( \pi(x) \) for a continuous random variable \( x \in \mathcal{X} \subset \mathbb{R}^D \), variational inference approximates \( \pi(x) \) with a tractable distribution, \( q(x; \lambda) \), from which we can efficiently draw samples and form sample-based estimates of functions of \( x \). Variational methods minimize the KL-divergence, \( \text{KL}(q || \pi) \), between \( q(\cdot; \lambda) \) and the true \( \pi \) as a function of variational parameter \( \lambda \) (Bishop, 2006). Although direct minimization of

\[\text{KL}(q || \pi) = \int q(x; \lambda) \log \frac{q(x; \lambda)}{\pi(x)} \, dx\]

We assume \( \pi(x) \) is known up to a constant, \( \tilde{\pi}(x) = C \pi(x) \).
We define our class of approximating distributions to be

\[ q(x; \lambda) = \sum_{c=1}^{C} \rho_c q_c(x; \lambda_c) \],

where we denote the full mixture as \( q \). This becomes probabilistic, which constructs a Markov chain such that the target distribution remains invariant. Expectations with respect to the target distribution can be calculated as an average with respect to these correlated samples. MCMC typically enjoys nice asymptotic properties; as the number of samples grows, MCMC samplers represent the true target distribution with increasing fidelity. However, rules for constructing correct Markov steps are restrictive. With a few exceptions, most MCMC algorithms require evaluating a log-likelihood that touches all data at each step in the chain (Maclaurin & Adams, 2014; Welling & Teh, 2011). This becomes problematic during statistical analyses of large amounts of data — MCMC is often considered unusable because of this computational bottleneck. Notably, variational methods can avoid this bottleneck by sub-sampling the data (Ranganath et al., 2016), as unbiased estimates of the log-likelihood can often be straight-forwardly used with optimization methods.

In the next section, we propose an algorithm that iteratively grows the approximating class \( Q \) and reframes the VI procedure as a series of optimization problems, resulting in a practical inference method that can both represent arbitrarily complex distributions and scale to large data sets.

### 3. Variational Boosting

We define our class of approximating distributions to be mixtures of \( C \) simpler component distributions:

\[ q^{(C)}(x; \lambda, \rho) = \sum_{c=1}^{C} \rho_c q_c(x; \lambda_c) \],

where we denote the full mixture as \( q^{(C)} \), mixing proportions \( \rho = (\rho_1, \ldots, \rho_C) \), and component distributions \( q_c(\cdot; \lambda_c) \) parameterized by \( \lambda = (\lambda_1, \ldots, \lambda_C) \). The \( q_c(\cdot; \lambda_c) \)'s can be any distribution over \( \mathcal{X} \subseteq \mathbb{R}^D \) from which we can efficiently draw samples using a continuous mapping parameterized by \( \lambda_c \) (e.g., multivariate normal (Jaakkola & Jordan, 1998), or a composition of invertible maps (Rezende & Mohamed, 2015)).

When posterior expectations and variances are of interest, mixture distributions provide tractable summaries. Expectations are easily expressed in terms of component expectations:

\[ E_{q^{(C)}}[f(x)] = \int q^{(C)}(x) f(x) dx = \sum_c \rho_c E_{q_c}[f(x)]. \]

In the case of multivariate normal components, the mean and covariance of a mixture are easy to compute, as are marginal distributions along any set of dimensions.

**Variational boosting (VBoost)** begins with a single mixture component, \( q^{(1)}(x; \lambda) = q_1(x; \lambda_1) \) with \( C = 1 \). We fix \( \rho_1 = 1 \) and use existing black-box variational inference methods to fit the first component parameter, \( \lambda_1 \). At the next iteration \( C = 2 \), we fix \( \lambda_1 \) and introduce a new component into the mixture, \( q_2(x; \lambda_2) \). We define a new ELBO objective as a function of new component parameters, \( \lambda_2 \), and a new mixture weight, \( \rho_2 \). We then optimize this objective with respect to \( \lambda_2 \) and \( \rho_2 \) until convergence. At each subsequent round, \( c \), we introduce new component parameters and a mixing weight, \( (\lambda_c, \rho_c) \), which are then optimized according to a new ELBO objective. The name **variational boosting** is inspired by methods that iteratively construct strong learners from ensembles of weak learners. We apply VBoost to target distributions via black-box variational inference with the re-parameterization trick to fit each component and mixture weights (Kingma & Welling, 2014; Ranganath et al., 2014; Salimans & Knowles, 2013). However, using mixtures as the variational approximation complicates the use of the re-parameterization trick.

### 3.1. The re-parameterization trick and mixtures

The re-parameterization trick is used to compute an unbiased estimate of the gradient of an objective that is expressed as an intractable expectation with respect to a continuous-valued random variable. This situation arises in variational inference when the ELBO cannot be evaluated analytically. We form an unbiased estimate as:

\[ \mathcal{L}(\lambda) = \mathbb{E}_q[\ln \pi(x) - \ln q(x; \lambda)] \]

\[ \approx \frac{1}{L} \sum_{\ell=1}^{L} \left[ \ln \pi(x^{(\ell)}) - \ln q(x^{(\ell)}; \lambda) \right] \]

where \( x^{(\ell)} \sim q(x; \lambda) \). To obtain a Monte Carlo estimate of the gradient of \( \mathcal{L}(\lambda) \) using the re-parameterization trick, we first separate the randomness needed to generate \( x^{(\ell)} \) from the parameters \( \lambda \), by defining a deterministic map \( x^{(\ell)} \triangleq f_q(\epsilon; \lambda) \) such that \( \epsilon \sim p(\epsilon) \) implies
The existing approx initial new comp target

When \( q(\cdot; \lambda) \) is a mixture, applying the re-parameterization trick is not straightforward. The typical sampling procedure for a mixture model includes a discrete random variable that indicates a mixture component, which complicates differentiation. We circumvent this by re-writing the variational objective as a weighted combination of expectations with respect to individual mixture components:

\[
\mathcal{L}(\lambda, \rho) = \int \left( \sum_{c=1}^{C} \rho_c q_c(x; \lambda_c) \right) \left[ \ln \pi(x) - \ln q(x; \lambda) \right] dx \\
= \sum_{c=1}^{C} \rho_c \int q_c(x; \lambda_c) \left[ \ln \pi(x) - \ln q(x; \lambda) \right] dx \\
= \sum_{c=1}^{C} \rho_c \mathbb{E}_{q_c}\left[ \ln \pi(x) - \ln q(x; \lambda) \right]
\]

which is a weighted sum of component-specific ELBOs. If the \( q_c \) are continuous and there exists some function \( f_c(\epsilon; \lambda) \) such that \( x = f_c(\epsilon; \lambda) \) and \( x \sim q_c(\cdot; \lambda) \) when \( \epsilon \sim p(\epsilon) \), then we can apply the re-parameterization trick to each component to obtain gradients of the ELBO:

\[
\nabla_{\lambda_c} \mathcal{L}(\lambda, \rho) = \nabla_{\lambda_c} \sum_{c=1}^{C} \rho_c \mathbb{E}_{x \sim q(x; \lambda)} \left[ \ln \pi(x) - \ln q(x; \lambda) \right] \\
= \sum_{c=1}^{C} \rho_c \mathbb{E}_{x \sim p(\epsilon)} \left[ \nabla_{\lambda_c} \ln q(f_c(\epsilon; \lambda_c)) - \nabla_{\lambda_c} \ln q(f_c(\epsilon; \lambda_c); \lambda) \right].
\]

VBoost leverages the above formulation of \( \nabla_{\lambda_c} \mathcal{L}(\lambda, \rho) \) to use the re-parameterization trick in a component-by-component manner, allowing us to improve the variational approximation as we incorporate new components.

3.2. Incorporating New Components

Next, we describe how to incrementally add components during the VBoost procedure.

The first component VBoost starts by fitting a approximation to \( \pi(x) \) consisting of a single component, \( q_1 \). We do this by maximizing the first ELBO objective

\[
\mathcal{L}^{(1)}(\lambda_1) = \mathbb{E}_{q_1}\left[ \ln \pi(x) - \ln q_1(x; \lambda_1) \right] \\
\lambda_1^* = \arg\max_{\lambda_1} \mathcal{L}^{(1)}(\lambda_1).
\]

Depending on the forms of \( \pi \) and \( q_1 \), optimizing \( \mathcal{L}^{(1)} \) can be accomplished by various methods — an obvious choice being black-box VI with the re-parameterization trick. After convergence we fix \( \lambda_1 \) to be \( \lambda_1^* \).

Adding a component to Eq. (5) introduces a new component parameter, \( \lambda_{C+1} \), and a new mixing weight, \( \rho_{C+1} \). In this section, the mixing parameter \( \rho_{C+1} \in [0, 1] \) mixes between the new component, \( q_{C+1}(\cdot; \lambda_{C+1}) \) and the existing approximation, \( q^{(C)} \). The new approximate distribution is

\[
q^{(C+1)}(x; \lambda, \rho) = \sum_{c=1}^{C} \rho_c q_c(x; \lambda_c) + (1 - \rho_{C+1}) q_{C+1}(x; \lambda_{C+1}).
\]

The new ELBO, as a function of \( \rho_{C+1} \) and \( \lambda_{C+1} \), is:

\[
\mathcal{L}^{(C+1)}(\rho_{C+1}, \lambda_{C+1}) = \mathbb{E}_{x \sim q^{(C+1)}}\left[ \ln \pi(x) - \ln q^{(C+1)}(x; \lambda_{C+1}, \rho_{C+1}) \right] \\
= (1 - \rho_{C+1}) \mathbb{E}_{q^{(C)}}\left[ \ln \pi(x) - \ln q^{(C+1)}(x; \lambda_{C+1}, \rho_{C+1}) \right] \\
+ \rho_{C+1} \mathbb{E}_{q_{C+1}}\left[ \ln \pi(x) - \ln q^{(C+1)}(x; \lambda_{C+1}, \rho_{C+1}) \right].
\]

Cruically, we have separated out two expectations: one with respect to the existing approximation, \( q^{(C)} \) (which is fixed), and the other with respect to the new component distribution, \( q_{C+1} \). Because we have fixed \( q^{(C)} \), we only need to optimize the new component parameters, \( \lambda_{C+1} \) and \( \rho_{C+1} \), allowing us to use the re-parameterization trick to obtain gradients of \( \mathcal{L}^{(C+1)} \). Note that evaluating the gradient requires sampling from the existing components which may result in larger variance than typical black-box variational methods. To mitigate the extra variance we use many samples to estimate the gradient and leave variance reduction to future work.
Figure 1 illustrates the algorithm on a simple one-dimensional example — the initialization of a new component and the resulting mixture after optimizing the second objective, $L^{(2)}(\rho_2, \lambda_2)$. Figure 2 depicts the result of VBoost on a two-dimensional, multi-modal target distribution. In both cases, the component distributions are Gaussians with diagonal covariance.

3.3. Structured Multivariate Normal Components

Though our method can use any component distribution that can be sampled using a continuous mapping, a sensible choice of component distribution is a multivariate normal

$$q(x; \lambda) = \mathcal{N}(x; \mu_\lambda, \Sigma_\lambda) = [2\pi|\Sigma_\lambda|]^{-1/2} \exp \left(-\frac{1}{2} (x - \mu_\lambda)^\top \Sigma^{-1}_\lambda (x - \mu_\lambda)\right)$$

where the variational parameter $\lambda$ is transformed into a mean vector $\mu_\lambda$ and covariance matrix $\Sigma_\lambda$.

Specifying the structure of the covariance matrix as a covariance that largely depends on the dimensionality of $X \subseteq \mathbb{R}^D$ and the correlation structure of the target distribution. A common choice of covariance is a diagonal matrix, $\Sigma_\lambda = \text{diag}(\sigma^2_1, \ldots, \sigma^2_D)$, which implies that $x$ is independent across dimensions. When the approximation only consists of one component, this structure is commonly referred to as the mean field family. While computationally efficient, mean field approximations cannot model posterior correlations, which often leads to underestimation of marginal variances. Additionally, when diagonal covariances are used as the component distributions in Eq. (5) the resulting mixture may require a large number of components to represent the strong correlations (see Fig. 2). Furthermore, independence constraints can actually introduce local optima in the variational objective (Wainwright & Jordan, 2008).

On the other end of the spectrum, we can parameterize the entire covariance matrix using the Cholesky decomposition, $L$, such that $LL^\top = \Sigma$. This allows $\Sigma$ to be any positive semi-definite matrix, enabling $q$ to have the full flexibility of a $D$-dimensional multivariate normal distribution. However, this introduces $D(D+1)/2$ parameters, which can be computationally cumbersome when $D$ is even moderately large. Furthermore, only a few pairs of variables may exhibit posterior correlations, particularly in multi-level models or neural networks where different parameter types may be nearly independent in the posterior.

As such, we would like to incorporate some capacity to capture correlations between dimensions of $x$ without overparameterizing the approximation. The next subsection discusses a covariance specification that provides this tradeoff, while remaining computationally tractable.

Low-rank plus diagonal covariance  Black-box variational inference methods with the re-parameterization trick require sampling from the variational distribution and efficiently computing (or approximating) the entropy of the variational distribution. For multivariate normal distributions, the entropy is a function of the determinant of the covariance matrix, $\Sigma$, while computing the log likelihood requires computing $\Sigma^{-1}$. When the dimensionality of the target, $D$, is large, computing determinants and inverses will have $O(D^3)$ time complexity and therefore may be prohibitively expensive to compute at every iteration.

However, it may be unnecessary to represent all $D(D-1)/2$ possible correlations in the target distribution, particularly if certain dimensions are close to independent. One way to increase the capacity of $q(x; \lambda)$ is to model the covariance as a low-rank plus diagonal (LR+D) matrix

$$\Sigma = FF^\top + \text{diag}(\exp(v))$$

where $F \in \mathbb{R}^{D \times r}$ is a matrix of off diagonal factors, and $v \in \mathbb{R}^D$ is the log-diagonal component. This is effectively approximating the target via a factor analysis model.

The choice of the rank $r$ presents a tradeoff: with a larger rank, the variational approximation can be more flexible; with a lower rank, the computations necessary for fitting the variational approximation are more efficient. As a concrete example, in Section 4 we present a $D = 37$ dimensional posterior resulting from a non-conjugate hierarchical model, and we show that a “rank $r = 2$ plus diagonal” covariance does an excellent job capturing all $D(D-1)/2 = 780$ pairwise correlations and $D$ marginal variances. Incorporating more components using the VBoost framework further improves the approximation of the distribution.

To use the re-parameterization trick with this low rank covariance, we can simulate from $q$ in two steps

$$z^{(lo)} \sim \mathcal{N}(0, I_r) \quad z^{(hi)} \sim \mathcal{N}(0, I_D)$$

$$x = Fz^{(lo)} + \mu + \mathcal{I}(v/2)z^{(hi)}$$
where \( z^{(bo)} \) generates the randomness due to the low-rank structure, and \( z^{(ha)} \) generates the randomness due to the diagonal structure. We use the operator \( \mathcal{I}(a) = \text{diag} (\exp(a)) \) for notational brevity. This generative process can be differentiated, yielding Monte Carlo estimates of the gradient with respect to \( F \) and \( v \) suitable for stochastic optimization.

In order to use LR+D covariance structure within VBoost, we will need to efficiently compute the determinant and inverse of \( \Sigma \). The matrix determinant lemma expresses the determinant of \( \Sigma \) as the product of two determinants

\[
\det (FF^T + \mathcal{I}(v)) = \det (\mathcal{I}(v)) \det (I_r + F^T \mathcal{I}(-v) F)
\]

where the left term is simply the product of the diagonal component, and the right term is the determinant of an \( r \times r \) matrix, computable in \( O(r^3) \) time (Harville, 1997).

To compute \( \Sigma^{-1} \), the Woodbury matrix identity states that

\[
(FF^T + \mathcal{I}(v))^{-1} = \mathcal{I}(-v) - \mathcal{I}(-v) F (I_r + F^T \mathcal{I}(-v) F)^{-1} F^T \mathcal{I}(-v)
\]

which involves the inversion of a smaller, \( r \times r \) matrix and can be done in \( O(r^3) \) time (Golub & Van Loan, 2013). Importantly, for \( r \ll D \) the above operations are efficiently differentiable and amenable for use in the BBVI framework.

**Fitting the rank** To specify the ELBO objective, we need to choose a rank \( r \) for the component covariance. There are many ways to decide on the rank of the variational approximation, some more appropriate for certain settings given dimensionality and computational constraints. For instance, we can greedily incorporate new rank components. Alternatively, we can fit a sequence of ranks \( r = 1, 2, \ldots, r_{\text{max}} \), and choose the best result (in terms of KL). In the Bayesian neural network model, we report results for a fixed schedule of ranks. In the hierarchical Poisson model, we monitor the change in marginal variances to decide the appropriate rank. See Section B of the supplement for further discussion.

**Initializing new component parameters** When we add a new component, we must first initialize the component parameters. We find that the VBoost optimization procedure can be sensitive to initialization, so we devise a cheap importance sampling-based algorithm to generate a good starting point. This initialization procedure is detailed in Section A and Algorithm 1 of the supplement.

### 3.4. Related Work

Mixtures of mean field approximations were introduced in Jaakkola & Jordan (1998) where mean field-like updates were developed using a bound on the entropy term and model-specific parameter updates. Nonparametric variational inference, introduced in Gershman et al. (2012), is a black-box variational inference algorithm that approximates a target distribution with a mixture of equally-weighted isotropic normals. The authors use a lower-bound on the entropy term in the ELBO to make the optimization procedure tractable. Similarly, Salimans & Knowles (2013) present a method for fitting mixture distributions as an approximation. However, their method is restricted to mixture component distributions within the exponential family, and a joint optimization procedure. Mixture distributions are a type of hierarchical variational model (Ranganath et al., 2016b), where the component identity can be thought of as latent variables in the variational distribution. While in Ranganath et al. (2016b), the authors optimize a lower bound on the ELBO to fit general hierarchical variational models, our approach integrates out the discrete latent variables, allowing us to directly optimize the ELBO.

Sequential maximum-likelihood estimation of mixture models has been studied previously where the error between the sequentially learned model and the optimal model where all components and weights are jointly learned is bounded by \( O(1/C) \) where \( C \) is the number of mixture components (Li & Barron, 1999; Li, 1999; Rakhlin et al., 2006). A similar bound was proven in Zhang (2003) using arguments from convex analysis. More recently, sequentially constructing a mixture of deep generative models has been shown to achieve the same \( O(1/C) \) error bound when trained using an adversarial approach (Tolstikhin et al., 2016). Though these ideas show promise for deriving error bounds for variational boosting, there are difficulties in applying them.

In concurrent work, Guo et al. (2016) developed a boosting procedure to construct flexible approximations to posterior distributions. In particular, they use gradient-boosting to determine candidate component distributions and then optimize the mixture weight for the new component (Friedman, 2000). However, Guo et al. (2016) assume that the gradient-boosting procedure is able to find the optimal new component so that the arguments in Zhang (2003) apply, which is not true in general. We note that if we make the similar assumption that at each step of VBoost the component parameters \( \lambda_i^* \) are found exactly, then the optimization of \( \rho_C \) is convex and can be optimized exactly. We can then appeal to the same arguments in Zhang (2003) and obtain an \( O(1/C) \) error bound. The work in Guo et al. (2016) provides important first steps in the theoretical development of boosting methods applied to variational inference, however, we note that developing a comprehensive theory that deals with the difficulties of multimodality and the non-joint-convexity of KL divergence in \( \lambda \) and \( \rho \) is still needed. Recently, Moore (2016) began to address issues of multimodality from model symmetry in variational inference. However, the question remains whether the entire distribution is being explored.
Seeger (2010) explored the use of low-rank covariance Gaussians as variational approximations using a PCA-like algorithm. Additionally, concurrent work has proposed the use a LR+D matrices as the covariances of Gaussian posterior approximations (Ong et al., 2017). We have also found that though the LR+D covariance approximation is useful for capturing posterior correlations, combining the idea with boosting new components to capture non-Gaussian posteriors yields superior posterior inferences.

4. Experiments and Analysis

To supplement the previous synthetic examples, we use VBoost to approximate various challenging posterior distributions arising from real statistical models of interest.\(^2\)

**Binomial Regression** We first apply VBoost to a non-conjugate hierarchical binomial regression model.\(^3\) The model describes the binomial rates of success (batting averages) of baseball players using a hierarchical model (Efron & Morris, 1975), parameterizing the “skill” of each player:

\[
\begin{align*}
\theta_j &\sim \text{Beta}(\phi \cdot \kappa, (1 - \phi) \cdot \kappa) \quad \text{player } j \text{ prior} \\
y_j &\sim \text{Binomial}(K_j, \theta_j) \quad \text{player } j \text{ hits ,}
\end{align*}
\]

where \(y_j\) is the number of successes (hits) player \(j\) has attempted in \(K_j\) attempts (at bats). Each player has a latent success rate \(\theta_j\), which is governed by two global variables \(\kappa\) and \(\phi\). We specify the priors \(\phi \sim \text{Unif}(0, 1)\) and \(\kappa \sim \text{Pareto}(1, 1.5)\). There are 18 players in this example, creating a posterior distribution with \(D = 20\) parameters. For each round of VBoost, we estimate \(\nabla_{\lambda, \rho} \mathcal{L}(c+1)\) using 400 samples each for \(q_{C+1}\) and \(q_C\). We use 1,000 iterations of adam with default parameters to update \(\rho_{C+1}\) and \(\lambda_{C+1}\) (Kingma & Ba, 2014).

In all experiments, we use autograd to obtain gradients with respect to new component parameters (Maclaurin et al., 2015b; a). To highlight the fidelity of our method, we compare VBoost with rank-1 components to mean field VI (MFVI) and the No-U-Turn Sampler (NUTS) (Hoffman & Gelman, 2014). The empirical distribution resulting from 20k NUTS samples is considered the “ground truth” posterior in this example. Figure 3a compares a selection of univariate and bivariate posterior marginals. We see that VBoost is able to closely match the NUTS posteriors, improving upon the MFVI approximation. Figure 3b compares the VBoost covariance estimates to the “ground truth” estimates of MCMC at various stages of the algorithm. We see that VBoost is able to capture pairwise covariances with increasing accuracy as the number of components increases.

**Multi-level Poisson GLM** We use VBoost to approximate the posterior of a hierarchical Poisson GLM, a common non-conjugate Bayesian model. Here, we focus on a specific model that was formulated to measure the relative rates of stop-and-frisk events for different ethnicities and in different precincts (Gelman et al., 2007), and has been used as an illustrative example of multi-level modeling (Gelman & Hill, 2006). The model uses a precinct and ethnicity effect to describe the relative rate of stop-and-frisk events

\[
\begin{align*}
\alpha_e &\sim \mathcal{N}(0, \sigma^2) \\
\beta_p &\sim \mathcal{N}(0, \sigma^2) \\
\ln \lambda_{ep} &= \mu + \alpha_e + \beta_p + \ln N_{ep} \\
Y_{ep} &\sim \mathcal{P}(\lambda_{ep})
\end{align*}
\]


\(^3\)Model and data from the mc-stan case studies.
As before, we simulate 20k NUTS samples, and compare various variational approximations. Because of the high non-Gaussian shape, most exhibited in the third column. Figures 6 and 7 in the supplement compare the presentation of this structure. As such, this example relies on the low-rank approximation to shape the posterior. Figure 4 shows how posterior accuracy is affected by incorporating covariance structure (left) and adding more components (right). Figures 6 and 7 in the supplement compare VBoost covariances to MCMC samples, showing that increased posterior rank capacity and number of components yield more accurate marginal variance and covariance estimates. These results indicate that while incorporating covariance structure increases the accuracy of estimating marginal variances, the non-Gaussianity afforded by the use of mixture components allows for a better posterior approximation translating into more accurate moment estimates.

**Bayesian Neural Network** We apply our method to a Bayesian neural network (BNN) regression model, which admits a high-dimensional, non-Gaussian posterior. We compare predictive performance of VBoost to Probabilistic Backpropagation (PBP) (Hernández-Lobato & Adams, 2015). Mimicking the experimental setup of Hernández-Lobato & Adams (2015), we use a single 50-unit hidden layer, with ReLU activation functions. We place a normal prior over each weight in the neural network, governed by the same variance and an inverse Gamma prior over the observation variance yielding the model:

\[
w_i \sim \mathcal{N}(0, 1/\alpha) \\
y|x, w, \tau \sim \mathcal{N}(\phi(x, w), 1/\tau)
\]

where \(w \in \{w_i\}\) is the set of weights, and \(\phi(x, w)\) is a multi-layer perceptron that maps input \(x\) to output \(y\) as a function of parameters \(w\). Both \(\alpha\) and \(\tau\) are given Gamma(1, 1) priors. We denote the set of parameters as \(\theta \triangleq (w, \alpha, \tau)\). We approximate the posterior \(p(w, \alpha, \tau|D)\),

| Table 1. Test log probability for PBP and VBoost with varying number of components (fixed rank of 5). Each entry shows the best average (though not necessarily “statistical significance”). |
|------------------|------------------|------------------|------------------|------------------|
| dataset          | php rank 5       | vboost 2         | vboost 6         | vboost 10        |
| wine             | -0.990 ± 0.08    | -0.972 ± 0.05    | -0.971 ± 0.05    | -0.978 ± 0.06    | -0.994 ± 0.06    |
| boston           | -2.902 ± 0.64    | -2.670 ± 0.16    | -2.651 ± 0.16    | -2.599 ± 0.16    | -2.628 ± 0.16    |
| concrete         | -3.162 ± 0.15    | -3.247 ± 0.06    | -3.228 ± 0.06    | -3.169 ± 0.07    | -3.134 ± 0.08    |
| power-plant      | -2.798 ± 0.04    | -2.814 ± 0.03    | -2.811 ± 0.03    | -2.800 ± 0.03    | -2.793 ± 0.03    |
| yacht            | -0.990 ± 0.08    | -0.972 ± 0.05    | -0.971 ± 0.05    | -0.978 ± 0.06    | -0.994 ± 0.06    |
| energy-eff.      | -1.971 ± 0.11    | -2.452 ± 0.12    | -2.422 ± 0.11    | -2.345 ± 0.11    | -2.299 ± 0.12    |

where \(Y_{ep}\) are the number of stop-and-frisk events within ethnicity group \(e\) and precinct \(p\) over some fixed period of time; \(N_{ep}\) is the total number of arrests of ethnicity group \(e\) in precinct \(p\) over the same period of time; \(\alpha_e\) and \(\beta_p\) are the ethnicity and precinct effects. The prior over the mean offset and group variances is given by \(\mu, \ln \sigma_{\alpha}^2, \ln \sigma_{\beta}^2 \sim \mathcal{N}(0, 10^2)\).

As before, we simulate 20k NUTS samples, and compare various variational approximations. Because of the high posterior correlations present in this example, VBoost with diagonal covariance components is inefficient in its representation of this structure. As such, this example relies on the low-rank approximation to shape the posterior. Figure 4 shows how posterior accuracy is affected by incorporating covariance structure (left) and adding more components (right). Figures 6 and 7 in the supplement compare VBoost covariances to MCMC samples, showing that increased posterior rank capacity and number of components yield more accurate marginal variance and covariance estimates. These results indicate that while incorporating covariance structure increases the accuracy of estimating marginal variances, the non-Gaussianity afforded by the use of mixture components allows for a better posterior approximation translating into more accurate moment estimates.
where $D$ is the training set of $\{(x_n, y_n)\}_{n=1}^{N}$ input-output pairs. We then use the posterior predictive distribution to compute the distribution for a new input $x^*$

$$p(y|x^*, D) = \int p(y|x^*, \theta)p(\theta|D)d\theta$$

$$\approx \frac{1}{L} \sum_{\ell=1}^{L} p(y|x^*, \theta^{(\ell)}), \theta^{(\ell)} \sim p(\theta|D)$$

and report average predictive log probabilities for held out data, $p(Y = y^*|x^*, D)$. For a dataset with input dimension $P$, the posterior has dimension $D = (P+2)\cdot 50+3$ (between $D = 303$ and $D = 753$ for the data sets considered).

We report held-out predictive performance for different approximate posteriors for six datasets. For each dataset, we perform the following training procedure 20 times. First, we create a random partition into a 90% training set and 10% testing set. We then apply VBoost, adding rank 5 components. We allow each additional component only 200 iterations. To save time on initialization, we draw 100 samples from the existing approximation, and initialize the new component with the sample with maximum weight. For comparison, Probabilistic back-propagation is given 1000 passes over the training data — empirically, sufficient for the algorithm to converge.

Table 3 in the supplement presents out-of-sample log probability for single-component multivariate Gaussian approximations with varying rank structure. Table 1 presents out-of-sample log probability for additional rank 5 components added using VBoost. We note that though we do not see much improvement as rank structure is added, we do see predictive improvement as components are added. Our results suggest that incorporating and adapting new mixture components is a recipe for a more expressive posterior approximation, translating into better predictive results. In fact, for all datasets we see that incorporating a new component improves test log probability, and we see further improvement with additional components for most of the datasets. Furthermore, in five of the datasets we see predictive performance surpass probabilistic back-propagation as new components are added. This highlights VBoost’s ability to trade computation for improved accuracy. These empirical results suggest that augmenting a Gaussian approximation to include additional capacity can improve predictive performance in a BNN while retaining computational tractability.

4.1. Comparison to NPVI

We also compare VBoost to nonparametric variational inference (NPVI) (Gershman et al., 2012), a similar mixture based black-box variational method. NPVI derives a tractable lower bound to the ELBO which is then approximately maximized. NPVI requires computing the Hessian approximately maximized. NPVI -718.47 -717.86 -717.09 -716.36 -715.86

Note that VBoost with 1 component is MFVI. All ELBO values are computed using a Monte Carlo estimate with $L = 100k$ samples from the variational distribution. In NPVI, each component is a spherical gaussian with a single $\sigma^2$ shared across all dimensions — this limits the capacity of the approximation, requiring more components. Note, VBoost greedily incorporates components, while NPVI is re-run using a different number of components.

We proposed VBoost, a practical variational inference method that constructs an increasingly expressive posterior approximation and is applicable to a variety of Bayesian models. We demonstrated the ability of VBoost to learn rich representations of complex, high-dimensional posteriors on a variety of real world statistical models. One avenue for future work is incorporating flexible component distributions such as compositions of invertible maps (Rezende & Mohamed, 2015) or auxiliary variable variational models (Maaløe et al., 2016). We also plan to study approximation guarantees of the VBoost method and variance reduction techniques for our reparameterization gradient approach. Also, when optimizing parameters of a variational family, recent work has shown that the natural gradient can be more robust and lead to better optima (Hoffman et al., 2013; Johnson et al., 2016). Deriving and applying natural gradient updates for mixture approximations could make VBoost more efficient.

Table 2. ELBO values for VBoost and NPVI (higher is better). Note that VBoost with 1 component is MFVI. All ELBO values are computed using a Monte Carlo estimate with $L = 100k$ samples from the variational distribution. In NPVI, each component is a spherical gaussian with a single $\sigma^2$ shared across all dimensions — this limits the capacity of the approximation, requiring more components. Note, VBoost greedily incorporates components, while NPVI is re-run using a different number of components.

5. Discussion and Conclusion

Table 2. ELBO values for VBoost and NPVI (higher is better). Note that VBoost with 1 component is MFVI. All ELBO values are computed using a Monte Carlo estimate with $L = 100k$ samples from the variational distribution. In NPVI, each component is a spherical gaussian with a single $\sigma^2$ shared across all dimensions — this limits the capacity of the approximation, requiring more components. Note, VBoost greedily incorporates components, while NPVI is re-run using a different number of components.
References


