NIASRA

NATIONAL INSTITUTE FOR APPLIED STATISTICS RESEARCH AUSTRALIA

National Institute for Applied Statistics Research Australia

University of Wollongong, Australia

Working Paper

09-21

Flexible Variational Bayes based on a Copula of a Mixture of Normals

David Gunawan, Robert Kohn, and David Nott

National Institute for Applied Statistics Research Australia, University of Wollongong, Wollongong NSW 2522, Australia Phone +61 2 4221 5435, Fax +61 2 4221 4998.
Email: karink@uow.edu.au

Flexible Variational Bayes based on a Copula of a Mixture of Normals

David Gunawan[1,3], Robert Kohn[2,3] and David Nott[4,5]

[1]School of Mathematics and Applied Statistics, University of Wollongong

[2]School of Economics, UNSW Business School, University of New South Wales

[3]Australian Center of Excellence for Mathematical and Statistical Frontiers

[4]Department of Statistics and Data Science, National University of Singapore

[5]Institute of Operations Research and Analytics, National University of Singapore

July 8, 2021

Abstract

Variational Bayes methods approximate the posterior density by a family of tractable distributions and use optimisation to estimate the unknown parameters of the approximation. Variational approximation is useful when exact inference is intractable or very costly. Our article develops a flexible variational approximation based on a copula of a mixture of normals, which is implemented using the natural gradient and a variance reduction method. The efficacy of the approach is illustrated by using simulated and real datasets to approximate multimodal, skewed and heavy-tailed posterior distributions, including an application to Bayesian deep feedforward neural network regression models. Each example shows that the proposed variational approximation is much more accurate than the corresponding Gaussian copula and a mixture of normals variational approximations.

Keywords: Natural-gradient; Non-Gaussian posterior; Multimodal; Stochastic gradient; Variance reduction

1 Introduction

Variational Bayes (VB) methods are increasingly used for Bayesian inference in a wide range of challenging statistical models (Ormerod and Wand, 2010; Blei et al., 2017). They formulate the problem of estimating the posterior density as an optimisation problem that approximates the target posterior density by a simpler and more tractable family of distributions; this family is usually selected to balance accuracy and computational cost. VB methods are usually computationally much cheaper than exact methods such as MCMC, particle MCMC, and sequential Monte Carlo (Del Moral et al., 2006). They are particularly useful in estimating the posterior densities of complex statistical models when exact inference is impossible or computationally expensive.

Much of the current literature focuses on Gaussian variational approximation (GVA) for approximating the target posterior density (Challis and Barber, 2013; Titsias and Lázaro-Gredilla, 2014; Kucukelbir et al., 2017; Tan and Nott, 2018; Ong et al., 2018). A key difficulty with the Gaussian approximation is that the number of covariance parameters increases quadratically with the number of model parameters. Diagonal approximations are one approach but lose the ability to model the dependence structure. Challis and Barber (2013) overcome this problem by considering a variety of sparse parametrizations of the Cholesky factor of the covariance matrix. Archer et al. (2016) and Tan and Nott (2018) parametrize the precision matrix in terms of its Cholesky factor and impose a sparsity structure on this factor that reflects any conditional independence structure in the model. A zero in the precision matrix of a Gaussian distribution corresponds to conditional independence between two model parameters, conditional on all the other parameters. Ong et al. (2018) use a sparse factor covariance structure to parsimoniously represent the covariance matrix in the Gaussian variational approximation.

Complex statistical models, such as Bayesian deep feedforward neural network (DFNN) regression models, usually exhibit skewed, multimodal, and heavy-tailed posterior distributions. Gaussian variational approximation for such models often poorly approximates such posterior distributions.

Han et al. (2016) and Smith et al. (2020) propose Gaussian copula variational approximations which are formed by using Gaussian distributions for element-wise parametric transformations of the target posterior. Smith et al. (2020) consider the Yeo-Johnson (Yeo and Johnson, 2000) and G&H families (Tukey, 1977) of element-

wise parametric transformations and use the sparse factor structures proposed by Ong et al. (2018) to model the covariance matrices of the Gaussian distributions. To fit the Gaussian copula variational approximation, they implement an efficient stochastic gradient ascent (SGA) optimisation method using the so called "reparameterisation trick" for obtaining unbiased estimates of the gradients of the variational objective function (Salimans and Knowles, 2013; Kingma and Welling, 2014; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014). The reparameterisation trick usually greatly reduces the variance of gradient estimates. We build on the Gaussian copula variational approximation and propose a Copula of a Mixture of Gaussians as a variational approximation (CMGVA), using sparse factor structures for the covariance matrices in the Gaussian mixture. The CMGVA allows fitting of multimodal, skewed, heavy tailed, and high-dimensional posterior distributions with complex dependence structures. We show in a number of examples that the CMGVA gives more accurate inference and forecasts than the corresponding Gaussian copula and mixture of normals approximation.

Variational optimisation of the CMGVA is challenging in complex models with a large number of parameters because of the large number of variational parameters that need to be optimised. The boosting variational inference method in Guo et al. (2016), and Miller et al. (2017) is a promising recent approach to fit mixture type variational approximations. By adding only a single mixture component at a time, the posterior approximation of the model parameters is iteratively refined. Miller et al. (2017) uses SGA optimisation with the reparameterisation trick to fit a a mixture of Gaussian densities with a factor covariance structure for the component densities. They found that the use of reparameterisation trick in the boosting variational method still results in a large variance and it is necessary to use many samples to estimate the gradient of the variational lower bound. Guo et al. (2016), Locatello et al. (2018) and Campbell and Li (2019) consider similar variational boosting mixture approximations, although they use different approaches to the optimisation and the specification of the mixture components. Jerfel et al. (2021) consider a boosting method which uses the forward KL-divergence and combines variational inference and importance sampling. We build upon the variational boosting method by efficiently adding a single mixture component at a time to fit the CMGVA; Section 4 gives further details. We also discuss efficient methods for performing the variational boosting optimisation method using the natural gradient (Amari, 1998) and the variance reduction method of Ranganath et al. (2014). Previous literature did not use the natural gradient in the variational boosting optimisation methods. Many natural-gradient variational methods have been recently proposed (Hoffman et al., 2013; Khan and Lin, 2017; Lin et al., 2019). This literature shows that natural-gradients

enable faster convergence than the traditional gradient-based methods because they exploit the information geometry of the variational approximation to speed-up the convergence of the optimisation.

The rest of the article is organised as follows. Section 2 gives background; Section 3 discusses the CMGVA; Section 4 discusses the variational optimisation algorithm that fits the CMGVA; Section 5 presents results from both simulated and real datasets; Section 6 concludes with a discussion of our approach and results.

2 Variational Inference

Let $\theta \in \Theta$ be the vector of model parameters, and $y_{1:n} = (y_1, ..., y_n)^\top$ the vector of observations. Bayesian inference about θ is based on the posterior distribution

$$\pi(\theta) = p(\theta|y_{1:n}) = \frac{p(y_{1:n}|\theta)\, p(\theta)}{p(y_{1:n})};$$

$p(\theta)$ is the prior, $p(y_{1:n}|\theta)$ is the likelihood function, and $p(y_{1:n})$ is the marginal likelihood. The posterior distribution $\pi(\theta)$ is unknown for most statistical models, making it challenging to carry out Bayesian inference. We consider variational inference methods, where a member $q_\lambda(\theta)$ of some family of tractable densities, indexed by the variational parameter $\lambda \in \Lambda$, is used to approximate the posterior $\pi(\theta)$. The optimal variational parameter λ is chosen by minimising the Kullback-Leibler divergence between $q_\lambda(\theta)$ and $\pi(\theta)$,

$$\begin{aligned} \mathrm{KL}(\lambda) &:= \int \log\left(\frac{q_\lambda(\theta)}{\pi(\theta)}\right) q_\lambda(\theta)\, d\theta. \\ &= \int q_\lambda(\theta) \log\left(\frac{q_\lambda(\theta)}{p(y_{1:n}|\theta)\, p(\theta)}\right) d\theta + \log p(y_{1:n}) \\ &= -\mathcal{L}(\lambda) + \log p(y_{1:n}), \end{aligned}$$

where

$$\mathcal{L}(\lambda) := \int \log\left(\frac{\pi(\theta)}{q_\lambda(\theta)}\right) q_\lambda(\theta)\, d\theta, \tag{1}$$

is a lower bound on the log of the marginal likelihood $\log p(y_{1:n})$. Therefore, minimising the KL divergence between $q_\lambda(\theta)$ and $\pi(\theta)$ is equivalent to maximising a variational lower bound (ELBO) on the log marginal likelihood $\log p(y_{1:n})$. The ELBO can be used as a tool for model selection.

The ELBO $\mathcal{L}(\lambda)$ in Eq. (1) is often an intractable integral with no closed form

solution. However, it can be written as an expectation with respect to $q_\lambda(\theta)$,

$$\mathcal{L}(\lambda) = E_{q_\lambda}\left(\log g(\theta) - \log q_\lambda(\theta)\right), \tag{2}$$

where $g(\theta) := p(y_{1:n}|\theta)\, p(\theta)$. This interpretation of Eq. (2) allows the use of stochastic gradient ascent (SGA) methods to maximise the variational lower bound $\mathcal{L}(\lambda)$. See, for e.g., Hoffman et al. (2013); Kingma and Welling (2014); Nott et al. (2012); Paisley et al. (2012); Salimans and Knowles (2013); Titsias and Lázaro-Gredilla (2014); Rezende et al. (2014). In SGA, an initial value $\lambda^{(0)}$ is updated according to the iterative scheme

$$\lambda^{(t+1)} := \lambda^{(t)} + a_t \circ \widehat{\nabla_\lambda \mathcal{L}(\lambda^{(t)})}, \quad \text{for } t = 0, 1, 2, ...; \tag{3}$$

$\circ$ denotes the Hadamard (element by element) product of two random vectors; $a_t := (a_{t1}, ..., a_{tm})^\top$ is a vector of step sizes and $\widehat{\nabla_\lambda \mathcal{L}(\lambda^{(t)})}$ is an unbiased estimate of the gradient of lower bound $\mathcal{L}(\lambda)$ at $\lambda = \lambda^{(t)}$. The learning rate sequence is chosen to satisfy the Robbins-Monro conditions $\sum_t a_t = \infty$ and $\sum_t a_t^2 < \infty$ (Robbins and Monro, 1951), which ensures that the iterates $\lambda^{(t)}$ converge to a local optimum as $t \to \infty$ under suitable regularity conditions (Bottou, 2010). Adaptive step sizes are often used in practice, and we employ the ADAM method of Kingma and Ba (2015), which uses bias-corrected estimates of the first and second moments of the stochastic gradients to compute adaptive learning rates. The update in Eq. (3) continues until a stopping criterion is satisfied.

To estimate the gradient, SGA methods often use the "log-derivative trick" (Kleijnen and Rubinstein, 1996), ($E_q(\nabla_\lambda \log q_\lambda(\theta)) = 0$), and it is straightforward to show that

$$\nabla_\lambda \mathcal{L}(\lambda) = E_q\left(\nabla_\lambda \log q_\lambda(\theta)\left(\log g(\theta) - \log q_\lambda(\theta)\right)\right); \tag{4}$$

E_q is the expectation with respect to $q_\lambda(\theta)$ in Eq. (4). Let

$$g_{\lambda_i} := \frac{1}{S}\sum_{s=1}^{S}\left(\log g(\theta_s) - \log q_\lambda(\theta_s)\right)\nabla_{\lambda_i} \log q_\lambda(\theta_s).$$

Then, $(g_{\lambda_1}, g_{\lambda_2}, ..., g_{\lambda_m})^\top$ is an unbiased estimate of $\nabla_\lambda \mathcal{L}(\lambda)$, where m is the dimension of the variational parameters λ. However, this approach usually results in large fluctuations in the stochastic gradients. Section 4 discusses variance reduction and natural gradient methods for obtaining the unbiased gradient estimates, which are very important for fast convergence and stability.

3 Flexible Copula of a Mixture of Gaussians Variational Approximations (CMGVA)

Smith et al. (2020) propose Gaussian copula variational approximations which are formed by using Gaussian distributions for an element-wise parametric transformations of the target posterior. They consider the Yeo-Johnson (Yeo and Johnson, 2000) and G&H families (Tukey, 1977) of element-wise parametric transformations and use the sparse factor structures proposed by Ong et al. (2018) as the covariance matrix of the Gaussian distributions. This section discusses the flexible CMGVA that builds on the Gaussian copula variational approximation of Smith et al. (2020).

Let $t_\gamma(\theta) := (t_{\gamma_1}(\theta_1), \ldots, t_{\gamma_m}(\theta_m))^\top$ be a family of one-to-one transformations with parameter vector γ. Each parameter θ_i is first transformed as $\varphi_i := t_{\gamma_i}(\theta_i)$; the density of $\varphi := (\varphi_1, \ldots, \varphi_m)^\top$ is then modeled as a K component multivariate mixture of normals. The variational approximation density for θ is then obtained by computing the Jacobian of the element-wise transformation from θ_i to φ_i, for $i = 1, ..., m$, such that

$$q_\lambda(\theta) := \sum_{k=1}^{K} \pi_k N(\varphi | \mu_k, \Sigma_k) \prod_{i=1}^{m} \dot{t}_{\gamma_i}(\theta_i); \tag{5}$$

the variational parameters are

$$\lambda = \left(\gamma_1^\top, ..., \gamma_m^\top, \left(\mu_1^\top, ..., \mu_K^\top\right), (\pi_1, ..., \pi_K), (\text{vech}(\Sigma_1), ..., \text{vech}(\Sigma_K))\right)^\top [1];$$

$\dot{t}_{\gamma_i}(\theta_i) := d\varphi_i / d\theta_i$ and $\sum_{k=1}^{K} \pi_k = 1$. The marginal densities of the approximation are

$$q_{\lambda_i}(\theta) = \sum_{k=1}^{K} \pi_k N(\varphi_i | \mu_{k,i}, \Sigma_{k,i}) \prod_{i=1}^{m} \dot{t}_{\gamma_i}(\theta_i), \tag{6}$$

for $i = 1, ..., m$, with $\lambda_i = \left(\gamma_i^\top, \left\{\mu_{k,i}^\top, \text{vech}(\Sigma_{k,i})\right\}_{k=1}^{K}, \{\pi_k\}_{k=1}^{K}\right)^\top$, a sub-vector of λ. As in Smith et al. (2020), the variational parameters λ are all identified in $q_\lambda(\theta)$ without additional constraints because they are also parameters of the margins given in Eq. (6). These variational approximations can fit multimodal, skewed, heavy tails, and high-dimensional posterior distributions with complex dependence structures.

When θ is high dimensional, we follow Ong et al. (2018) and adopt a factor structure for each Σ_k, $k = 1, ..., K$. Let β_k be an $m \times r_k$ matrix, with $r_k \ll m$. For

[1] For an $m \times m$ matrix A, vec (A) is the vector of length m^2 obtained by stacking the columns of A under each other from left to right; vech (A) is the vector of length $m(m+1)/2$ obtained from vec (A) by removing all the superdiagonal elements of the matrix A.

identifiability, we set the strict upper triangle of β_k to zero. Let $d_k = (d_{k,1}, ..., d_{k,m})^\top$ be a parameter vector with $d_{k,i} > 0$, and denote by D_k the $m \times m$ diagonal matrix with entries d_k. We assume that

$$\Sigma_k := \beta_k \beta_k^\top + D_k^2,$$

so that the number of parameters in Σ_k grows linearly with m if $r_k \ll m$ is kept fixed. Note that the number of factors r_k can be different for each component of the mixture for $k = 1, ..., K$.

To draw S samples from the variational approximation given in Eq. (5), the indicator variables G_s, for $s = 1, ..., S$, are first generated; each G_s selects the component of the mixture from which the sample is to be drawn, with $G_s = k$ with probability π_k. Then, $(z_{G_s,s}, \eta_{G_s,s}) \sim N(0, I)$ are generated, where $z_{G_s,s}$ is r_{G_s}-dimensional and $\eta_{G_s,s}$ is m-dimensional; $\varphi_s = \mu_{G_s} + \beta_{G_s} z_{G_s,s} + d_{G_s} \circ \eta_{G_s,s}$ are then calculated, where $\circ$ denotes the Hadamard product defined above. This representation shows that the latent variables z, which are low-dimensional, explain all the correlation between the transformed parameters φ_s, and the parameter-specific idiosyncratic variance is captured by η. Finally, $\theta_{s,i} = t_{\gamma_i}^{-1}(\varphi_{s,i})$ is generated for $i = 1, ..., m$ and $s = 1, ..., S$. The Yeo-Johnson (YJ) transformation (Yeo and Johnson, 2000) is used for the transformation t_{γ_i} for all $i = 1, ..., m$. If a parameter θ_i is constrained, it is first transformed to the real line; for example, with a variance parameter we transform θ_i to its logarithm. Smith et al. (2020) gives the inverses and derivatives with respect to the model and parameters. Both the Gaussian copula and mixture of normals variational approximations are special cases of the CMGVA. The mixture of normals variational approximation is a special case of the CMGVA when the variational parameters $\gamma_i = 1$, for all $i = 1, ..., m$. The Gaussian copula is a special case of the CMGVA when the number of component in the mixture is $K = 1$.

4 Variational Methods

This section discusses the estimation method for the flexible variational approximation in Eq. (5). We first describe how the first component of the mixture is fitted and then the process for adding an additional component to the existing mixture approximation.

4.1 Optimisation Methods

The method starts by fitting an approximation to the posterior distribution $\pi(\theta)$ with a single mixture distribution, $K = 1$, using the variational optimisation algorithm given in Smith et al. (2020); the optimal first component variational parameters are denoted as $\lambda_1^* := \left(\mu_1^\top, \beta_1^\top, d_1^\top, \pi_1, \gamma\right)^\top$, with the mixture weight π_1 set to 1. We do this by maximising the first lower bound objective function

$$\mathcal{L}^{(1)}(\lambda_1) = E_{q_\lambda}\left(\log g(\theta) - \log q_{\lambda_1}^{(1)}(\theta)\right), \quad \lambda_1^* = \arg\max_{\lambda_1} \mathcal{L}^{(1)}(\lambda_1),$$

where

$$q_{\lambda_1}^{(1)}(\theta) = N\left(\varphi; \mu_1, \beta_1\beta_1^\top + D_1^2\right) \prod_{i=1}^{m} \dot{t}_{\gamma_i}(\theta_i).$$

After the optimisation algorithm converges, λ_1 is fixed as λ_1^*.

After iteration K, the current approximation to the posterior distribution $\pi(\theta)$ is a mixture distribution with K components

$$q_\lambda^{(K)}(\theta) = \sum_{k=1}^{K} \pi_k N\left(\varphi|\mu_k, \beta_k\beta_k^\top + D_k^2\right) \prod_{i=1}^{m} \dot{t}_{\gamma_i}(\theta_i).$$

We can introduce a new mixture component with new component parameters, $(\mu_{K+1}, \beta_{K+1}, d_{K+1})$, and a new mixing weight π_{K+1}. The weight $\pi_{K+1} \in [0, 1]$ mixes between the new component and the existing approximation. The new approximate distribution is

$$q_\lambda^{(K+1)}(\theta) := \left((1-\pi_{K+1})\left(\sum_{k=1}^{K} \pi_k N\left(\varphi|\mu_k, \beta_k\beta_k^\top + D_k^2\right)\right) + \pi_{K+1} N\left(\varphi|\mu_{K+1}, \beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)\right) \prod_{i=1}^{m} \dot{t}_{\gamma_i}(\theta_i).$$

The new lower bound objective function is

$$\begin{aligned} \mathcal{L}^{(K+1)}(\lambda_{K+1}) &:= E_{q_\lambda}\left(\log g(\theta) - \log q_\lambda^{(K+1)}(\theta)\right), \\ \lambda_{K+1}^* &:= \arg\max_{\lambda_{K+1}} \mathcal{L}^{(K+1)}(\lambda_{K+1}). \end{aligned}$$

Note that with the copula transformation fixed, updating the mixture approximation parameters is the same as updating an ordinary mixture approximation in the transformed space of φ. Since the existing variational approximation is also fixed, it is only necessary to optimise the new component parameters $(\mu_{K+1}, \beta_{K+1}, d_{K+1})$,

and the new mixing weight π_{K+1}, which reduces the dimension of the variational parameters to be optimised. Although the existing mixture components are fixed, their mixing weights can vary. It is possible to reoptimise the variational parameters γ_i for all $i = 1, ..., m$ at each iteration of the algorithm. However, we obtained no substantial improvement with the increased computational cost. The γ_i, for all $i = 1, ..., m$, are kept fixed for iteration $k > 1$. The next section discusses how to update the variational parameters.

4.2 Updating the Variational Parameters

This section outlines the updating scheme for the variational parameters of the new component parameters $(\mu_{K+1}, \beta_{K+1}, D_{K+1})$ and the new mixing weight π_{K+1} based on natural-gradient methods and control-variates for reducing the variance of the unbiased estimates of the gradient of the lower bound. Many natural-gradient methods for variational inference have been proposed recently (Hoffman et al., 2013; Khan and Lin, 2017); these show that natural-gradients enable faster convergence than traditional gradient-based methods.

The natural-gradient approach exploits the information geometry of the variational approximation q_λ to speed-up the convergence of the optimisation. Assuming that the Fisher information matrix (FIM), denoted by F_λ, of the $q_\lambda(\theta)$ is positive-definite for all $\lambda \in \Lambda$, the natural-gradient update is

$$\lambda^{(t+1)} = \lambda^{(t)} + a_t \circ \left(F_\lambda^{-1} \widehat{\nabla_\lambda \mathcal{L}(\lambda^{(t)})} \right), \quad \text{for } t = 1, 2, ... \tag{7}$$

Multiplying the gradient of the lower bound by the inverse of the Fisher information matrix leads to a proper scaling of the gradient in each dimension and takes into account dependencies between variational parameters λ. The natural-gradient update in Eq. (7) requires computing and inverting the FIM, which can be computationally expensive in high-dimensional problems. However, Khan and Nielsen (2018) shows that the natural-gradient update can be much simpler than the traditional gradient-based methods for exponential family (EF) variational approximations. The standard EF variational approximation is

$$q_\lambda(\theta) = h(\theta) \exp\left[\langle \phi(\theta), \lambda \rangle - A(\lambda) \right],$$

where $\phi(\theta)$ is the sufficient statistic, $h(\theta)$ is the base measure, and $A(\lambda)$ is the log-partition. For such approximations, it is unnecessary to compute the Fisher information matrix (FIM) explicitly and the expectation parameter $m_\theta(\lambda) = \mathbb{E}_q(\phi(\theta))$ can be used to compute natural-gradients, provided the FIM is invertible. The update

for the natural-gradient methods is now

$$\lambda^{(t+1)} = \lambda^{(t)} + a_t \circ \left(\widehat{\nabla_{m_\theta} \mathcal{L}\left(\lambda^{(t)}\right)} \right), \quad \text{for } t = 1, 2, ... \tag{8}$$

The following relationship is used to get the results in Eq. (8):

$$\nabla_\lambda \mathcal{L}\left(\lambda\right) = \left[\nabla_\lambda m_\theta^\top\right] \nabla_{m_\theta} \mathcal{L}\left(\lambda\right) = \left[F_\lambda\right] \nabla_{m_\theta} \mathcal{L}\left(\lambda\right);$$

the first equality is obtained by using the chain rule and the second equality is obtained by noting $\nabla_\lambda m_\theta^\top = \nabla_\lambda^2 A\left(\lambda\right) = F_\lambda$; see Lin et al. (2019) for a more detailed discussion.

Lin et al. (2019) derive natural gradient methods for a mixture of EF distributions in the conditional exponential family form

$$q_\lambda\left(\theta, \underline{w}\right) = q_\lambda\left(\theta | \underline{w}\right) q_\lambda\left(\underline{w}\right), \tag{9}$$

with $q_\lambda\left(\theta|\underline{w}\right)$ as the component and $q_\lambda\left(\underline{w}\right)$ as the mixing distribution. A special case is the finite mixture of Gaussians, where the components in EF form are mixed using a multinomial distribution. They show that if the FIM, $F_\lambda\left(\theta, \underline{w}\right)$, of the joint distribution of θ and $\underline{w}$, is invertible, then it is possible to derive natural-gradient update without explicitly computing the FIM. They use the following update

$$\lambda^{(t+1)} := \lambda^{(t)} + a_t \circ \left(\widehat{\nabla_{m} \mathcal{L}\left(\lambda^{(t)}\right)} \right), \quad \text{for } t = 1, 2, ...,$$

with the expectation parameters $m := (m_\theta, m_{\underline{w}})$, where $m_\theta := E_{q_\lambda(\theta|\underline{w}) q_\lambda(\underline{w})}\left(\phi\left(\theta, \underline{w}\right)\right)$, with[2] $\phi\left(\theta, \underline{w}\right) := \left\{ \mathrm{I}_k\left(\underline{w}\right)\theta, \mathrm{I}_k\left(\underline{w}\right)\theta\theta^\top \right\}_{k=1}^{K-1}$, and $m_w := E_{q_\lambda(\underline{w})}\left(\phi\left(\underline{w}\right)\right)$ with $\phi(w) = \left\{\mathrm{I}_k\left(\underline{w}\right)\right\}_{k=1}^{K-1}$. This results in simple natural-gradient updates for the mixture components and weights. As a factor structure is used for the covariance matrix, we adopt the natural-gradient updates of Lin et al. (2019) only for the new weight π_{K+1} and the mixture means μ_{K+1}. Denote $\pi_1' := \left(1 - \pi_{K+1}\right) \sum_{k=1}^K \pi_k$ and $\pi_2' := \pi_{K+1}$, $\pi_1' + \pi_2' = 1$. The natural-gradient update for the new mixture weights, π_{K+1} is

$$\log\left(\frac{\pi_1'}{\pi_2'}\right)^{(t+1)} = \log\left(\frac{\pi_1'}{\pi_2'}\right)^{(t)} - a_t\left(\delta_1 - \delta_2\right)\left(\log\left(\pi\left(\theta\right)\right) - \log q_\lambda^{(K+1)}\left(\theta\right)\right), \tag{10}$$

where $q_\lambda^{(K+1)}(\theta)$ is given in Eq. (5) and the natural-gradient update for the new means μ_{K+1} is

$$\mu_{K+1}^{(t+1)} = \mu_{K+1}^{(t)} - a_t \delta_2 \left(\beta_{K+1}^{(t)} \beta_{K+1}^{(t)^\top} + D_{K+1}^{2(t)}\right)\left(\nabla_\theta \log\left(\pi\left(\theta\right)\right) - \nabla_\theta \log q_\lambda^{(K+1)}\left(\theta\right)\right); \tag{11}$$

[2] $\mathrm{I}_k(\underline{w})$ denotes the indicator function which is 1 if $\underline{w} = k$, and 0 otherwise

$$\delta_1 = \left(\sum_{k=1}^{K} \pi_k N\left(\varphi|\mu_k, \beta_k\beta_k^\top + D_k^2\right)\right)/\delta_{tot}, \quad \delta_2 = \left(N\left(\varphi|\mu_{K+1}, \beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)\right)/\delta_{tot},$$

where

$$\delta_{tot} = \left((1-\pi_{K+1})\left(\sum_{k=1}^{K} \pi_k N\left(\varphi|\mu_k, \beta_k\beta_k^\top + D_k^2\right)\right) + \pi_{K+1} N\left(\varphi|\mu_{K+1}, \beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)\right).$$

Updating the variational parameters β_{K+1} and d_{K+1} is now discussed. There are two reasons why the reparameterisation trick is not used to update the variational parameters β_{K+1} and d_{K+1}. The first is that Miller et al. (2017) found that using the reparameterisation trick in the boosting variational method still results in a large variance and it is necessary to use many samples to estimate the gradient of the variational lower bound. The second is that it may not be possible to use the reparameterisation trick because of the copula transformation. An alternative method to update the variational parameters β_{K+1} and d_{K+1} is now discussed.

The gradients of the lower bound in Eq. (4) require the gradient $\nabla_\lambda \log q_\lambda(\theta)$. The gradient of $\log q_\lambda(\theta)$ with respect to β_{K+1} is

$$\nabla_{\text{vech}(\beta_{K+1})} \log q_\lambda(\theta) = \frac{\pi_{K+1} N\left(\varphi|\mu_{K+1}, \beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)}{\delta_{tot}} \text{vech}\Big(-\left(\beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)^{-1}\beta_{K+1} + \left(\beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)^{-1}(\varphi-\mu_{K+1})(\varphi-\mu_{K+1})^\top\left(\beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)^{-1}\beta_{K+1}\Big),$$

and the gradient of $\log q_\lambda(\theta)$ with respect to d_{K+1} is

$$\nabla_{d_{K+1}} \log q_\lambda(\theta) = \frac{\pi_{K+1} N\left(\varphi|\mu_{K+1}, \beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)}{\delta_{tot}} \text{diag}\Big(-\left(\beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)^{-1} D_{K+1} + \left(\beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)^{-1}(\varphi-\mu_{K+1})(\varphi-\mu_{K+1})^\top\left(\beta_{K+1}\beta_{K+1}^\top + D_{K+1}^2\right)^{-1} D_{K+1}\Big).$$

We also employ control variates as in Ranganath et al. (2014) to reduce the variance of an unbiased estimate of gradient of the $\nabla_{\text{vech}(\beta_{K+1})}\mathcal{L}(\lambda)$ and $\nabla_{d_{K+1}}\mathcal{L}(\lambda)$ and the efficient natural-gradient updates using the conjugate gradient methods given in Tran et al. (2019). To use a conjugate gradient linear solver to compute $F_\lambda^{-1}\nabla_\lambda\mathcal{L}(\lambda)$ it is only necessary to be able to quickly compute matrix vector products of the form $F_\lambda x$ for any vector x, without needing to store the elements of F_λ. When β_{K+1} is a vector, the natural gradient can be computed efficiently as outlined in Algorithm

1. Our updates for β_{K+1} and d_{K+1} are pre-conditioned gradient steps based on what a natural gradient update would be for a Gaussian approximation in the added component, not the natural gradient in the mixture approximation. However, we find this pre-conditioned gradient step improves efficiency compared to the ordinary gradient. Algorithm 2 gives the full variational algorithm.

Algorithm 1 Computing the natural gradients for the vectors β_{K+1} and d_{K+1}.

Input: vectors β_{K+1} and d_{K+1} and the standard gradients $g_{\beta_{K+1}} = \nabla_{vech(\beta_{K+1})}\mathcal{L}(\lambda)$ and $g_{d_{K+1}} = \nabla_{d_{K+1}}\mathcal{L}(\lambda)$

Output: $g^{nat}_{\beta_{K+1}} = F_\lambda^{-1} g_{\beta_{K+1}} = \nabla^{nat}_{vech(\beta_{K+1})}\mathcal{L}(\lambda)$ and $g^{nat}_{d_{K+1}} = F_\lambda^{-1} g_{d_{K+1}} = \nabla^{nat}_{d_{K+1}}\mathcal{L}(\lambda)$.

- Compute the vectors: $v_1 = d^2_{K+1} - 2\beta^2_{K+1} \circ d^{-4}_{K+1}$, $v_2 = \beta^2_{K+1} \circ d^{-3}_{K+1}$, and the scalars $\kappa_1 = \sum_{i=1}^m \beta^2_{K+1}/d^2_{K+1}$, and $\kappa_2 = 0.5\left(1 + \sum_{i=1}^m v^2_{2i}/v_{1i}\right)^{-1}$.
- Compute:

$$g^{nat}_{\beta_{K+1}} = \frac{1+\kappa_1}{2\kappa_1}\left(\left(g^\top_{\beta_{K+1}}\beta_{K+1}\right)\beta_{K+1} + d^2_{K+1} \circ g_{\beta_{K+1}}\right), \tag{12}$$

and

$$g^{nat}_{d_{K+1}} = 0.5 v_1^{-1} \circ g_{d_{K+1}} + \kappa_2\left[\left(v_1^{-1} \circ v_2\right)^\top g_{d_{K+1}}\right]\left(v_1^{-1} \circ v_2\right). \tag{13}$$

Algorithm 2 Variational Algorithm

1. (a) Initialize $\lambda_{K+1}^{(0)} = \left(\mu_{K+1}^{\top(0)}, \text{vech}(\beta_{K+1}^{(0)}), d_{K+1}^{\top(0)}, \pi_{K+1}^{(0)}\right)$, set $t = 0$, and generate $\theta_s^{(t)} \sim q_\lambda^{(K+1)}(\theta)$ for $s = 1, ..., S$. Let m_β, m_d, be the number of elements in vech (β_{K+1}), and d_{K+1}.

   (b) Evaluate the control variates $\varsigma_{vech(\beta_{K+1})}^{(t)} = \left(\varsigma_{1,vech(\beta_{K+1})}^{(t)}, ..., \varsigma_{m_\beta,vech(\beta_{K+1})}^{(t)}\right)'$, $\varsigma_{d_{K+1}}^{(t)} = \left(\varsigma_{1,d_{K+1}}^{(t)}, ..., \varsigma_{m_d,d_{K+1}}^{(t)}\right)'$, with

$$\varsigma_{i,d_{K+1}}^{(t)} = \frac{\text{cov}\left(\left[\log\left(\pi\left(\theta\right)\right) - \log q_\lambda\left(\theta\right)\right]\nabla_{\lambda_{i,d_{K+1}}}\log q_\lambda\left(\theta\right), \nabla_{\lambda_{i,d_{K+1}}}\log q_\lambda\left(\theta\right)\right)}{\mathbb{V}\left(\nabla_{\lambda_{i,d_{K+1}}}\log q_\lambda\left(\theta\right)\right)}, \tag{14}$$

   for $i = 1, ..., m_d$, where cov and $\mathbb{V}(\cdot)$ are the sample estimates of covariance and variance based on S samples from step (1a). The $\varsigma_{vech(\beta_{K+1})}^{(t)}$ can be estimated similarly.

Repeat until the stopping rule is satisfied

- Update β_{K+1}, d_{K+1}:

1. Generate $\theta_s^{(t)} \sim q_\lambda^{(K+1)}(\theta)$ for $s = 1, ..., S$.
2. Compute $\widehat{\nabla_{\text{vech}(\beta_{K+1})}\mathcal{L}}\left(\lambda^{(t)}\right) = \left(g_{1,\text{vech}(\beta_{K+1})}^{(t)}, ..., g_{m_\beta,\text{vech}(\beta_{K+1})}^{(t)}\right)$, with

$$g_{i,\text{vech}(\beta_{K+1})}^{(t)} = \frac{1}{S}\sum_{s=1}^{S}\left[\log\left(\pi\left(\theta_s^{(t)}\right)\right) - \log q_\lambda\left(\theta_s^{(t)}\right) - \varsigma_{i,\text{vech}(\beta_{K+1})}^{(k)}\right]\nabla_{\text{vech}(\beta_{K+1})}\log q_\lambda\left(\theta_s\right) \tag{15}$$

3. The gradient of lower bound $\widehat{\nabla_{d_{K+1}}\mathcal{L}}\left(\lambda^{(t)}\right) = \left(g_{1,d_{K+1}}^{(t)}, ..., g_{m_d,d_{K+1}}^{(t)}\right)$ can be computed similarly as in Eq. (15).
4. Compute the control variate $\varsigma_{\text{vech}(\beta_{K+1})}^{(t)}$ and $\varsigma_{d_{K+1}}^{(t)}$ as in Eq. (14) and compute $g_{\beta_{K+1}}^{nat}$ and $g_{d_{K+1}}^{nat}$ using algorithm 1.
5. Compute $\triangle d_{K+1}$, $\triangle\text{vech}\left(\beta_{K+1}\right)$ using ADAM methods described in Section 4.4. Then, set $d_{K+1}^{(t+1)} = d_{K+1}^{(t)} + \triangle d_{K+1}$, $\text{vech}\left(\beta_{K+1}\right)^{(t+1)} = \text{vech}\left(\beta_{K+1}\right)^{(t)} + \triangle\text{vech}\left(\beta_{K+1}\right)$.

- Update μ_{K+1} and π_{K+1}

1. Generate $\theta_s^{(t)} \sim q_\lambda^{(K+1)}(\theta)$ for $s = 1, ..., S$.
2. Use Eq. (10) to update π_{K+1} and Eq. (11) to update μ_{K+1}, respectively. Set $t = t + 1$

4.3 Initialising a New Mixture Component

Introducing a new component requires setting the initial values for the new component parameters $(\mu_{K+1}, \beta_{K+1}, D_{K+1})$ and new mixing weight π_{K+1}. A good initial value for the new mixture component should be located in the region of the target posterior distribution $\pi(\theta)$ that is not well represented by the existing mixture approximation $q_\lambda^{(K)}(\theta)$. There are many ways to set the initial value. We discuss some suggestions that work well in our examples. The elements in β_{K+1} are initialized by $N(0, 0.01^2)$, the diagonal elements in D_{K+1} are initialized by 0.01, and the mixture weight π_{K+1} is initialized by 0.5. Algorithm (3) gives the initial value for $\mu_{K+1} = (\mu_{1,K+1}, ..., \mu_{m,K+1})^\top$.

Algorithm 3 Initial values for μ_{K+1}.

Input: $\{\pi_k, \mu_k, \beta_k, d_k\}_{k=1}^K$ and γ
Output: initial values for μ_{K+1}

- Compute the mean estimates of the current approximation $\widehat{\varphi}=\sum_{k=1}^K \pi_k \mu_k$.
- For each $i = 1$ to m
  - Construct a grid of S values of φ_i, fixing the other parameters at their posterior means or some other reasonable values, and denote this $m \times S$ matrix as φ^*. One way to construct the grid of S values is to draw samples from the current approximation, and compute the minimum and maximum values $(\min(\varphi_i), \max(\varphi_i))$ for φ_i for $i = 1, ..., m$.
  - Compute $\theta_s^* = t_\gamma^{-1}(\varphi_s^*)$ for $s = 1, ..., S$.
  - Compute the $w_s^* = \frac{\pi(\theta_s^*)}{q_\lambda^{(K)}(\theta_s^*)}$ for $s = 1, ..., S$
  - Set $\mu_{i,K+1} = \varphi_{i,s}^*$ with the largest weight, or choose $\mu_{i,K+1} = \varphi_{i,s}^*$ with a probability proportional to the weight w_s^*.
- Alternatively, when the dimension of the parameters is large,
  - Draw S samples from the current approximation φ_s, and compute $\theta_s^* = t_\gamma^{-1}(\varphi_s^*)$, and the weights $w_s^* = \pi(\theta_s^*)/q_\lambda^{(K)}(\theta_s^*)$ for $s = 1, ..., S$. Then, set $\mu_{K+1} = \varphi_s^*$ with the largest weight or choose $\mu_{K+1} = \varphi_s^*$ with a probability proportional to the weight w_s^*.

4.4 Learning Rate

Setting the learning rate in a stochastic gradient algorithm is very challenging, especially when the parameter vector is high dimensional. The choice of learning rate affects both the rate of convergence and the quality of the optimum attained. Learning rates that are too high can cause unstable optimisation, while learning rates

that are too low result in slow convergence and can lead to a situation where the parameters erroneously appear to have converged. In all our examples, the learning rates are set adaptively using the ADAM method (Kingma and Ba, 2015) that gives different step sizes for each element of the variational parameters λ. At iteration $t+1$, the variational parameter λ is updated as

$$\lambda^{(t+1)} := \lambda^{(t)} + \triangle^{(t)}.$$

Let g_t^{nat} denote the natural stochastic gradient estimate at iteration t. ADAM computes (biased) first and second moment estimates of the gradients using exponential moving averages,

$$\begin{aligned} m_t &= \tau_1 m_{t-1} + (1-\tau_1)\, g_t^{nat}, \\ v_t &= \tau_2 v_{t-1} + (1-\tau_2)\left(g_t^2\right)^{nat}, \end{aligned}$$

where $\tau_1, \tau_2 \in [0,1)$ control the decay rates. The biased first and second moment estimates can be corrected by

$$\widehat{m}_t = m_t / \left(1-\tau_1^t\right), \quad \widehat{v}_t = v_t / \left(1-\tau_2^t\right);$$

the change $\triangle^{(t)}$ is then computed as

$$\triangle^{(t)} = \frac{\alpha \widehat{m}_t}{\sqrt{\widehat{v}_t} + \epsilon}.$$

We set $\tau_1 = 0.9$, $\tau_2 = 0.99$, and $\epsilon = 10^{-8}$ (Kingma and Ba, 2015). It is possible to use different α for μ, β, d, and π. We set $\alpha_\mu = 0.01$, $\alpha_\beta = \alpha_d = \alpha_\pi = 0.001$, unless stated otherwise.

5 Examples

To illustrate the performance of the CMGVA, we employ it to approximate complex and high-dimensional posterior distributions of the model parameters for simulated and real datasets, where its greater flexibility may increase the accuracy of inference and prediction compared to simpler approximations.

This section has four examples. In the first example, the CMGVA is used to approximate a high dimensional, skewed, and heavy tailed distribution. In the second example, the CMGVA is used to approximate a high dimensional multimodal distribution. The third example uses the CMGVA to approximate the posterior

distributions of logistic and linear regression models with complex prior distributions. The fourth example uses the CMGVA to fit a Bayesian deep neural network regression model.

In all examples, the CMGVA is compared to the state-of-the-art Gaussian copula variational approximation of Smith et al. (2020) and mixture of normals variational approximations. All the examples are implemented in Matlab and run on a standard desktop computer. Unless otherwise stated, we use the estimates of the variational lower bound to select the optimal number of components in the mixture approximations. In principle, making the variational family more flexible by adding new components should not reduce the variational lower bound; in practice, the difficulty of the optimization means that adding a new component may worsen the approximation. The boosting approach, where an existing approximation is improved allows us to tune the accuracy/computational effort trade-off, where we start with a rough fast approximation and keep improving until the computational budget is exhausted. All the variational parameters are initialised using the approach in Section 4.3.

5.1 Skewed and Heavy-Tailed High-Dimensional Distributions

This section investigates the performance of the CMGVA to approximate skewed, heavy-tailed, and high-dimensional target distributions and compare it to Gaussian copula and mixture of normals variational approximations. The target distributions $\pi(\theta)$ are assumed to follow multivariate t-copula,

$$\pi(\theta) := \pi(\zeta) \prod_{i=1}^{m} \left| \frac{d\zeta_i}{d\theta_i} \right|, \tag{16}$$

for $i = 1, ..., m$. The density $\pi(\zeta)$ is a multivariate t-distribution with zero mean, full-covariance matrix (ones on the diagonal and 0.8 on the off-diagonals), and degrees of freedom $df = 4$. The Yeo-Johnson (YJ) transformation with parameters set to 0.5 (Yeo and Johnson, 2000) is used. The dimension of the parameters θ is set to $m = 100$. The number of factors r_1 is set to 4 for the first component and $r_k = 1$, for each additional mixture components for $k = 2, ..., 20$. We use $S = 50$ samples to estimate the gradients of the lower bound. The algorithm in Smith et al. (2020) is performed for 20000 iterations to obtain the optimal variational parameters for the first component of the mixture, and then Algorithm 2 is performed for 5000 iterations to obtain the optimal variational parameters for each additional component of the mixture.

We first compare the performance of the natural gradient method of updating μ,

β, and d to the ordinary gradient method. The mixture weights are still updated using the natural gradient method as described in Section 4.2. In this example, the step sizes $\alpha_\mu = \alpha_\beta = \alpha_d = 0.01$ and $\alpha_\pi = 0.001$, for the natural gradient method and $\alpha_\mu = \alpha_\beta = \alpha_d = \alpha_\pi = 0.001$ for the ordinary gradient method. The ordinary gradient is unstable and fails to converge with larger step sizes. Figure 1 shows the lower bound values over iterations for the ordinary gradient and natural gradient methods for the 2-component CMGVA for the multivariate t-copula example. The figure clearly shows that the natural gradient is much less noisy and converges much faster than the ordinary gradient. We therefore use the natural gradient method in all the examples reported below.

Figure 2 monitors the convergence of the CMGVA from $K = 2$ to 10 components via the estimated value of the lower bound. The figure shows the fast and stable convergence of the optimisation algorithm for all components. Figure 3 shows the average lower bound value over the last 500 steps of the optimisation algorithm for 2 to 20-components CMGVA. Comparing the values of the lower bound, it can be seen that there is substantial improvement obtained from $k = 3$ to 7 components compared to the Gaussian copula variational approximation, and there are further smaller improvements after that. The optimal number of components for this example is 17. The figure also shows that the Gaussian copula is better than the mixture of normals variational approximation for any number of components for this example.

Figure 4 shows the kernel density estimates of the marginal densities of the parameter θ_1 estimated using the Gaussian copula and CMGVA with $k = 2, .., 20$ components. Clearly, the CMGVA with the optimal number of components captures both the skewness and the heavy tails of the marginal posteriors $\pi(\theta_1)$ very well, whereas the Gaussian copula approximation does worse. Figure 5 shows the scatter plot of the observations from the first and second margins generated from the true target densities, Gaussian copula variational approximation, and the 15-components CMGVA. This confirms the observation that the CMGVA is much better at capturing the skewness and heavy-tailed properties of the true target densities compared to the Gaussian copula and mixture of normals variational approximations.

Figure 1: The plot of the lower bound values over iterations for the ordinary gradient and natural gradient methods for the 2-components CMGVA for the 100-dimensional multivariate t-copula example.

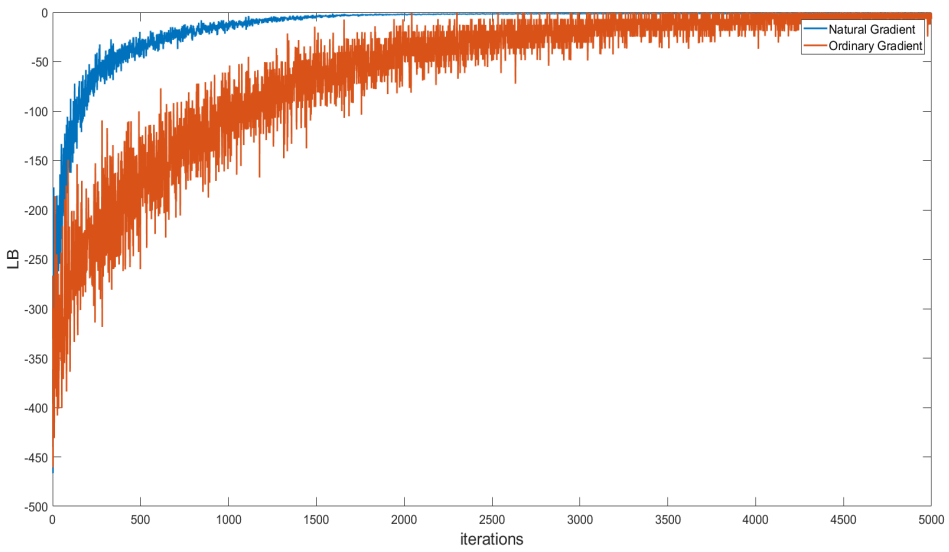


Figure 2: The plot of the lower bound values for the 2-components to 10-components CMGVA for the 100-dimensional multivariate t-copula example.

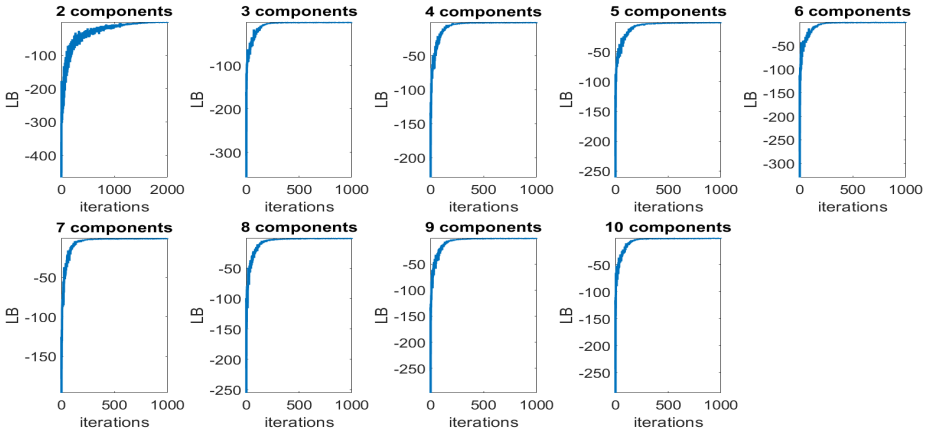


Figure 3: The plot of the average lower bound values over the last 500 steps for the CMGVA and mixture of normals variational approximation for the 100-dimensional multivariate t-copula example.

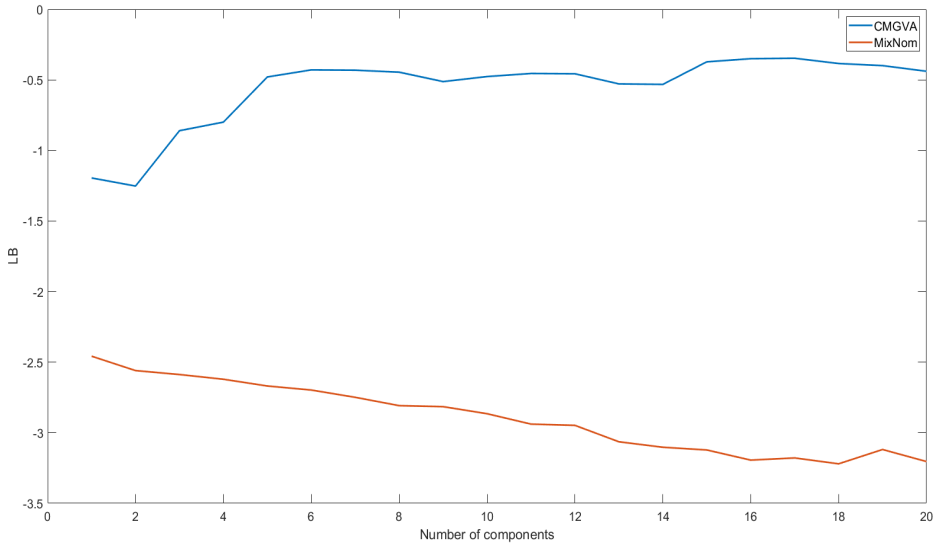


Figure 4: Kernel Density Estimates of the marginal parameter θ_1 approximated by Gaussian copula (red) and CMGVA (yellow) with true target density (blue).

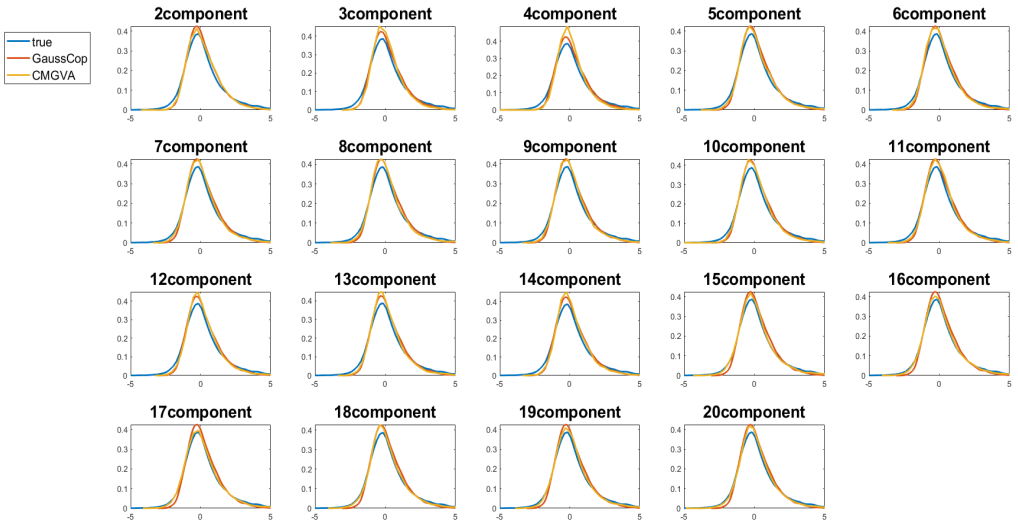
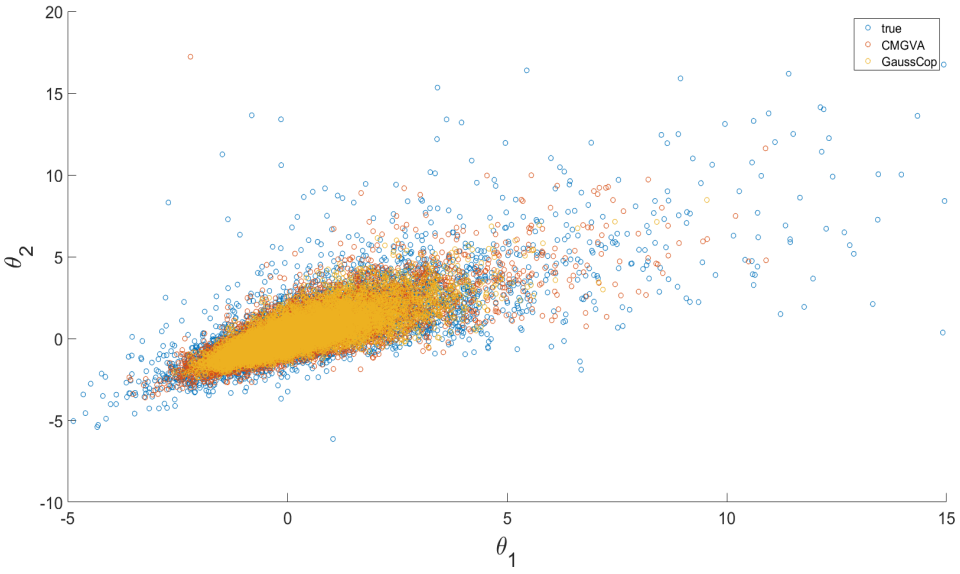


Figure 5: Scatter plot of the observations from the first and second margins generated from true target density (blue), Gaussian copula variational approximation (yellow), and 17-components CMGVA (orange).



5.2 Multimodal High-Dimensional Target Distributions

This section investigates the ability of the CMGVA to approximate multimodal high-dimensional target distributions and compares it to the Gaussian copula and the mixture of normals variational approximations. The target distribution is the multivariate mixture of normals,

$$\pi(\theta) = \sum_{c=1}^{3} w_c N(\theta | u_c, \Sigma_c).$$

The dimension of the parameters θ is set to 100. Each element $u_{i,c}$ is uniformly drawn from the interval $[-2, 2]$ for $i = 1, ..., 100$ and $c = 1, .., 3$. We set the full covariance matrix Σ_c with ones on the diagonal and the correlation coefficients $\rho = 0.2$ and 0.8 in the off-diagonals for all components. The number of factors $r_1 = 4$ for the first component and $r_k = 1$, for each additional mixture component for $k = 2, ..., 20$. Similarly to the previous example, we use $S = 50$ samples to estimate the gradients of the lower bound. The algorithm in Smith et al. (2020) is performed for 20000 iterations to obtain the optimal variational parameters for the first component of the mixture and then Algorithm 2 is performed for 5000 iterations to obtain the optimal variational parameters for each additional component of the mixture. The CMGVA is compared to the Gaussian copula and mixture of normals variational approximations. The step sizes are set to the values given in Section 5.1. Figure 6 shows the average

lower bound values over the last 500 steps for the CMGVA and mixture of normals variational approximation for the 100-dimensional mixture of normals example. The figure shows that the performance of the CMGVA is comparable to the mixture of normals variational approximation for this example. Figure 7 confirms that by showing that all the YJ-parameters are close to 1. The mixture of normals is a special case of CMGVA when all the YJ-parameters are equal to 1, and is clearly much better than the Gaussian copula in this example. Figure 8 monitors the convergence of the CMGVA from $K = 2$ to 10 components via the estimated value of the lower bound ($\rho = 0.8$). The figure clearly shows fast and stable convergence of the optimisation algorithm for all components even for high-dimensional target distribution. Similar conclusions can be made for the case $\rho = 0.2$.

Figures 9 and 10 show the kernel density estimates of some of the marginal parameters θ approximated with the Gaussian Copula, the optimal CMGVA, and the optimal mixture of normals for the cases $\rho = 0.2$ and 0.8, respectively, together with the true marginal distributions. Clearly, the CMGVA and mixture of normals approximate the true target densities of the marginal parameters θ much better than the Gaussian copula and are able to approximate multimodal posteriors. Finally, Figures 11 and 12 show the scatter plots of the observations from the 10th and 20th margins generated from the true target density, Gaussian copula variational approximation, and the optimal CMGVA for $\rho = 0.2$ and 0.8, respectively. These figures also confirm that both the CMGVA and mixture of normals capture the bimodality and complex-shaped of the two dimensional target distribution of the parameters.

We now study the performance of CMGVA as an approximation to the target distributions which are q-variate normal with zero mean and full covariance matrix with ones on the diagonal, and with correlation coefficients $\rho = 0.8$ in the off-diagonals. Figure 13 plots the average lower bound values over the last 500 steps for the CMGVA for this example and shows that there is no improvement obtained by adding additional components in the mixture.

The two examples suggest that: (1) The CMGVA is able to approximate heavy tails, multimodality, skewness and other complex properties of the high dimensional targets very well on a variety of examples, and is better than the Gaussian copula or a mixture of normals variational approximations. The Gaussian copula is better than a mixture of normals at approximating a skewed target distribution as in Section 5.1. The mixture of normals is clearly better than a Gaussian copula at approximating multimodal target distributions. (2) Adding a few components to the Gaussian copula improves its ability to approximate complex target distributions. Therefore, the proposed approach can be considered as a refinement of the Gaussian copula varia-

tional approximation. (3) The variational optimisation algorithm with the natural gradient approaches provides stable and fast convergence compared to ordinary gradient methods. (4) Adding components one at a time provides a practical variational inference method that constructs an increasingly complicated approximation and is applicable to a variety of multivariate target distributions.

Figure 6: The plot of the average lower bound values over the last 500 steps for the CMGVA and mixture of normals variational approximation for the 100-dimensional mixture of normals example.

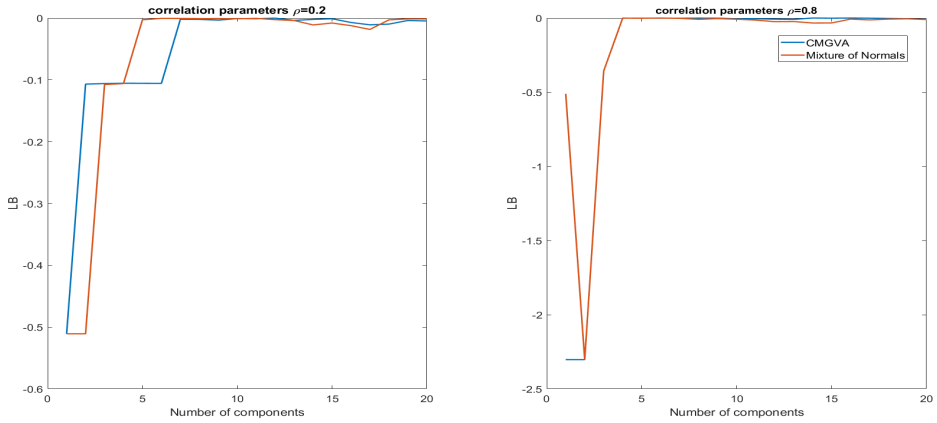


Figure 7: The plot of the YJ-parameters γ_i for all $i = 1, ..., m$ of the CMGVA for the mixture of normals example.

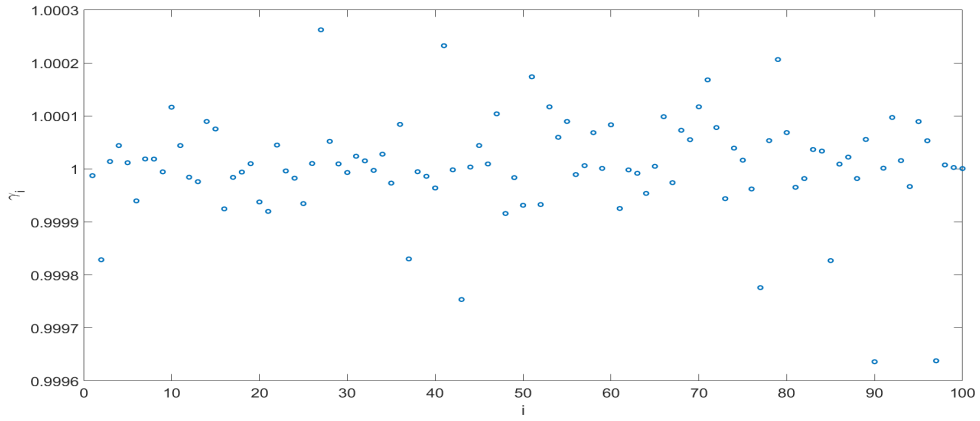


Figure 8: The plot of lower bound values for CMGVA for the 100-dimensional multivariate mixture of normals distribution examples with $\rho = 0.8$.

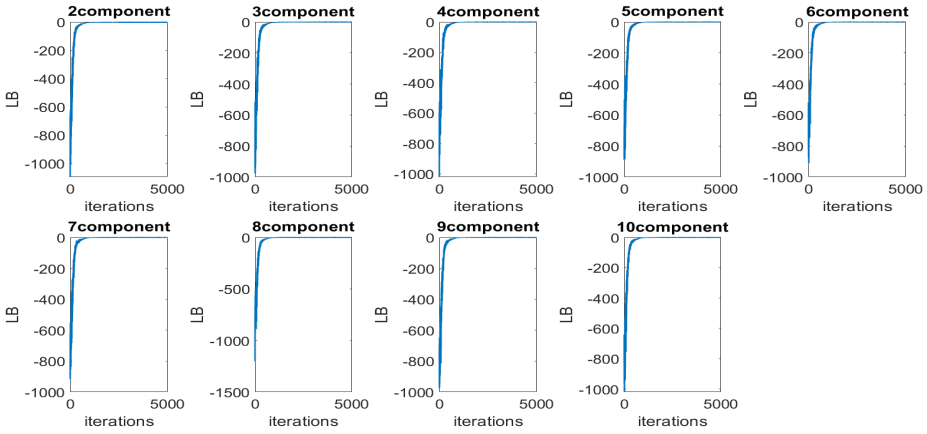


Figure 9: Kernel density estimates of some of the marginal parameters θ approximated with a Gaussian Copula, CMGVA (with the optimal number of components $K = 10$), and a mixture of normals variational approximation (with the optimal number of components $K = 6$) for the mixture of normals example with $\rho = 0.2$

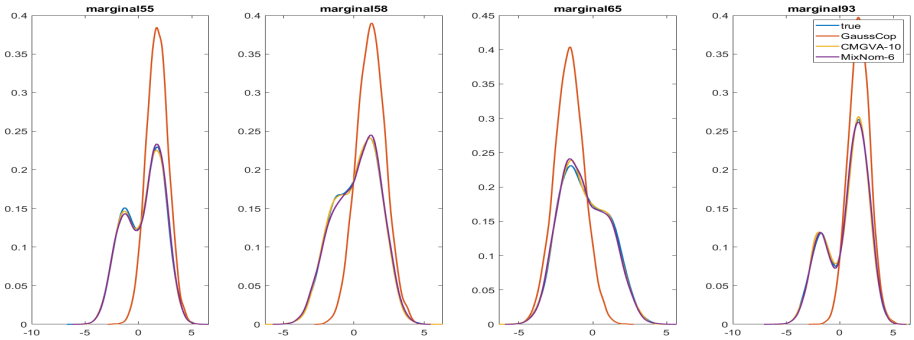


Figure 10: Kernel density estimates of some of the marginal parameters θ approximated with the Gaussian Copula, CMGVA (with the optimal number of components $K = 4$), and mixture of normals variational approximation (with the optimal number of components $K = 4$) for the mixture of normals example with $\rho = 0.8$

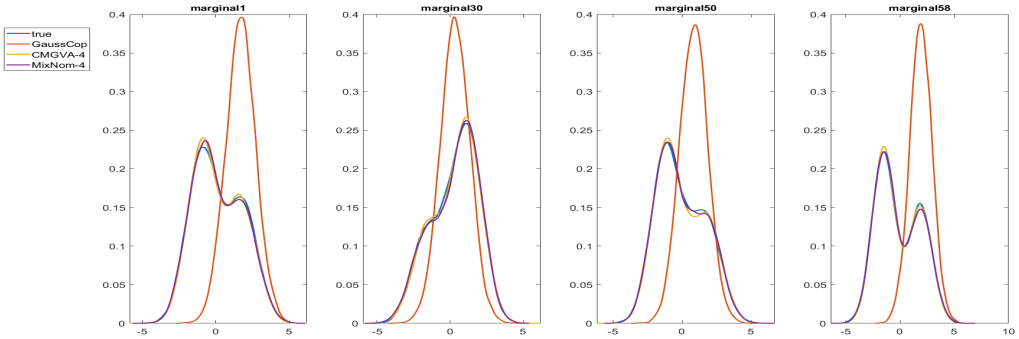


Figure 11: Top: Scatter plot of the observations generated from true target distribution (blue), and a 10-component CMGVA (orange) for the mixture of normals example with $\rho = 0.2$; Bottom: Scatter plot of the observations generated from true target distribution (blue), and Gaussian copula (orange) for the mixture of normals example with $\rho = 0.2$

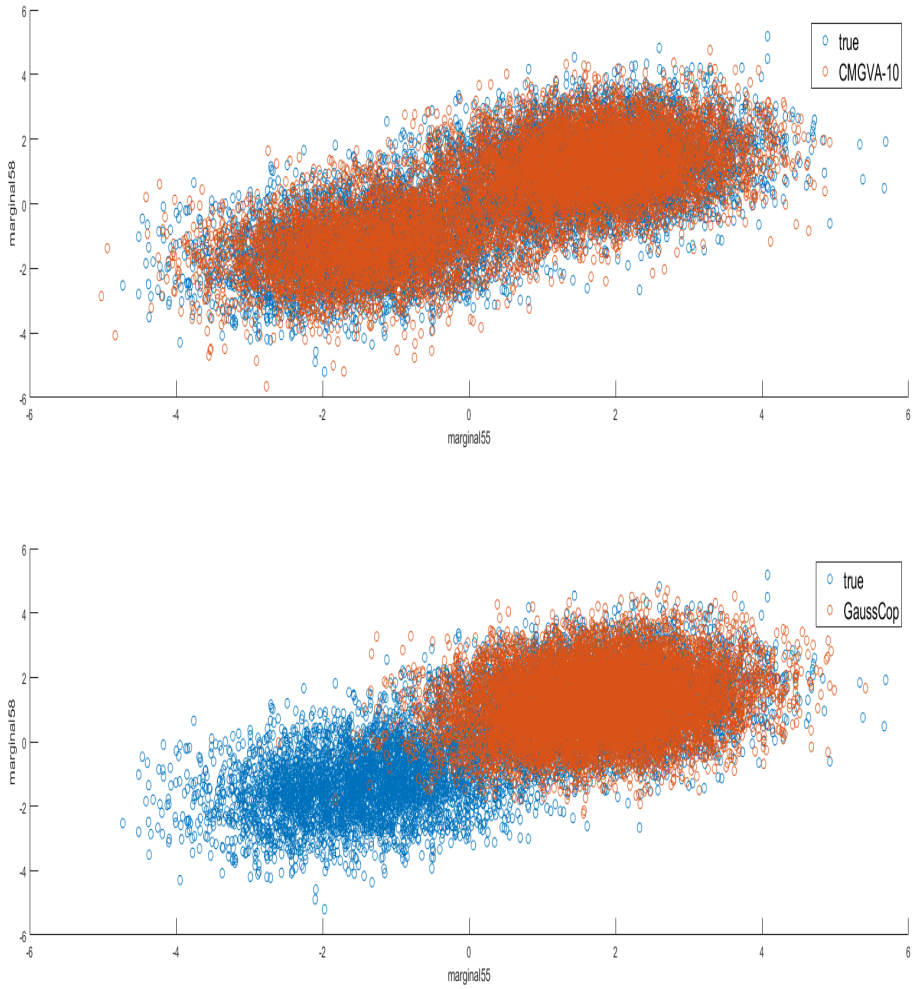


Figure 12: Top: Scatter plot of the observations generated from true target distribution (blue), and 4-component CMGVA (orange) for the mixture of normals example with $\rho = 0.8$; Bottom: Scatter plot of the observations generated from true target distribution (blue), and Gaussian copula (orange) for the mixture of normals example with $\rho = 0.8$

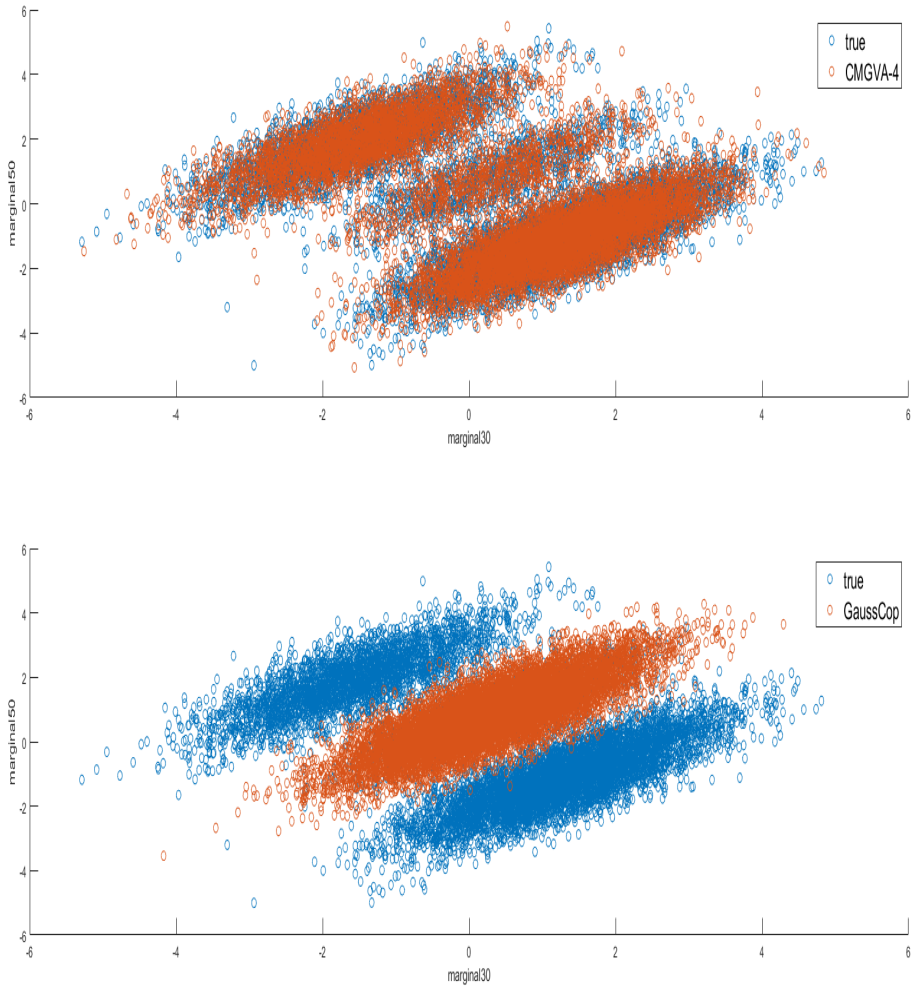
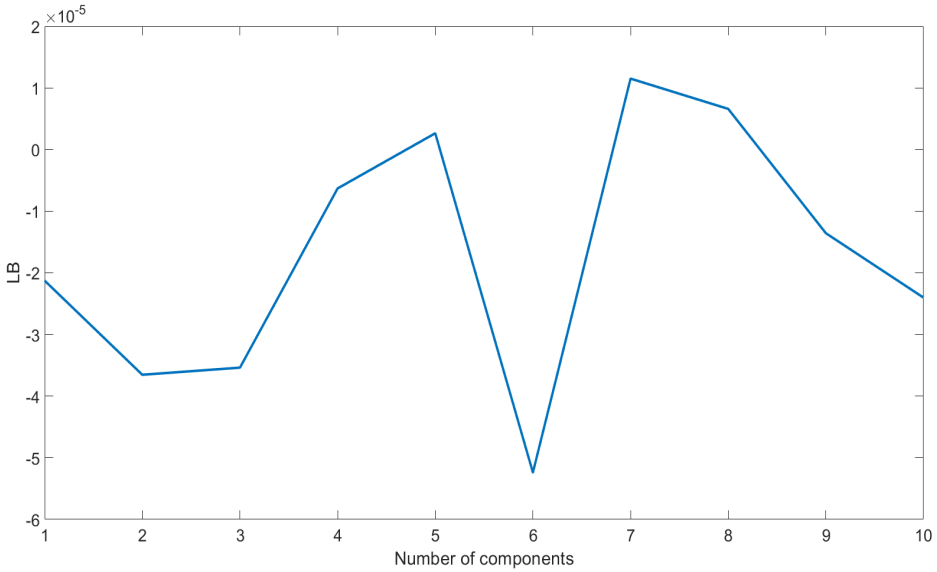


Figure 13: The plot of the average lower bound values over the last 500 steps for the CMGVA for the 100-dimensional normal dsitribution example.



5.3 Bayesian Regression Models with Complex Prior Distributions

This section considers linear and logistic regression models with complex prior distributions for the regression parameters $b = (b_0, b_1, ..., b_p)^\top$, where $p+1$ is the number of regression parameters. The prior for each regression parameter b_i, for $i = 1, ..., p$, is the mixture of skew normals of Azzalini (1985)

$$\begin{aligned} p\left(b_i|w, \mu_1, \sigma_1^2, \alpha_1, \mu_2, \sigma_2^2, \alpha_2\right) &= 2w\frac{1}{\sigma_1}\phi\left(\frac{b_i-\mu_1}{\sigma_1}\right)\Phi\left(\alpha_1\left(\frac{b_i-\mu_1}{\sigma_1}\right)\right) + \\ & \quad 2\left(1-w\right)\frac{1}{\sigma_2}\phi\left(\frac{b_i-\mu_2}{\sigma_2}\right)\Phi\left(\alpha_2\left(\frac{b_i-\mu_2}{\sigma_2}\right)\right), \end{aligned} \tag{17}$$

where ϕ and Φ are the standard normal density function and standard normal cdf, respectively. The prior for the b_0 is $N(0,1)$. The prior for the variance parameter τ^2 in the linear regression model is Gamma$(1,1)$; it is the gamma distribution with shape parameter 1 and scale parameter 1. The hyperparameters of the prior distribution in Eq. (17) are set to $\alpha_1 = \alpha_2 = -20$, $\sigma_1^2 = \sigma_2^2 = 1$, $\mu_1 = 0$, $\mu_2 = 1$, and $w = 0.5$. Figure 14 plots the prior distribution for the regression parameter b_i which exhibits both skewness and multimodality. The complex prior distribution for each regression parameter is chosen to create a complex prior structure to evaluate the performance of CMGVA and compare it to Gaussian copula and mixture of normals variational approximations.

For logistic regression, the model for the response $y_i \in \{0, 1\}$, given $p\times 1$ covariates

and parameters, is

$$p\left(y_i|x_i,b\right)=\frac{\exp\left(y_i x_i^\top b\right)}{1+\exp\left(y_i x_i^\top b\right)}.$$

For linear regression, the model for the response $y_i \in R$, given a set of covariates and parameters is $p\left(y_i|x_i,b,\tau^2\right)=\frac{1}{\tau}\phi\left(\frac{y_i-x_i^\top b}{\tau}\right)$. There are $p+1$ and $p+2$ parameters in the logistic and linear regression models, respectively.

We consider the spam, krkp, ionosphere, and mushroom data for the logistic regressions; they have sample sizes $n = 4601$, 351, 3196, and 8124, with 104, 111, 37, and 95 covariates, respectively and are also considered by Ong et al. (2018) and Smith et al. (2020); the data are available from the UCI Machine Learning Repository (Lichman, 2013). [3] In the results reported below we only use the first 250 observations of each dataset. The small dataset size and complex prior distribution for the regression parameters b are chosen to create a complex posterior structure.

For the Bayesian linear regression models, we consider two datasets: a direct marketing dataset and an abalone dataset. The direct marketing dataset is originally from the statistics textbook by Jank (2011); it consists of 1000 observations, with the response being the amount (in \$1000) a customer spends on the company's products per year. There are 11 covariates including gender, income, the number of catalogs sent, married status, young, old, etc; we include the interaction terms of all the covariates, so the total number of covariates is 67. The abalone dataset, available on the UCI Machine Learning Repository, has 4177 observations. The response variable is the number of rings used to determine the age of the abalone. There are 9 covariates including sex, length, diameter, as well as other measurements of the abalone. We include the interaction terms of all the covariates, giving a total number of covariates of 46. Only the first 50 observations from both datasets are used to estimate the Bayesian linear regression model. The small dataset size and complex prior distribution for the regression parameters b are chosen to create a complex posterior structure to evaluate the performance of CMGVA and compare it to the Gaussian copula and mixture of normals variational approximations. Similarly to the previous example, we set the number of factors r_1 to 4 for the first component and $r_k = 1$, for each additional mixture component for $k = 2, ..., 20$. We use $S = 50$ samples to estimate the gradients of the lower bound. The algorithm given in Smith et al. (2020) is performed for 20000 iterations to obtain the optimal variational parameters for the first component of the mixture and then the algorithm 2 is performed for 5000 iterations to obtain the optimal variational parameters for each additional component of the mixture.

Figures 15 and 16 show the average lower bound values over the last 500 steps

[3] see https://archive.ics.uci.edu/ml/datasets.php for further details.

of the optimisation algorithm for the CMGVA and mixture of normals variational approximation for the Bayesian logistic regression model for the four datasets. The figures show that the CMGVA is better than the mixture of normals variational approximation for any number of components. There is substantial improvement obtained by adding 3 to 8 components to the Gaussian copula variational approximation, and there are further small improvements after that for the ionosphere, krkp, and spam datasets. However, the lower bound values slightly decrease as more components are added for the mushroom dataset.

Figure 17 shows the average lower bound values over the last 500 steps of the optimisation algorithm for the CMGVA and mixture of normals variational approximation for the Bayesian linear regression model for the abalone and direct marketing datasets. The figure confirms that the CMGVA is better than the mixture of normals variational approximation for any number of components and there is substantial improvements obtained just by adding one or two components to the Gaussian copula variational approximation for both datasets.

Figure 14: The prior distribution for the regression parameters b

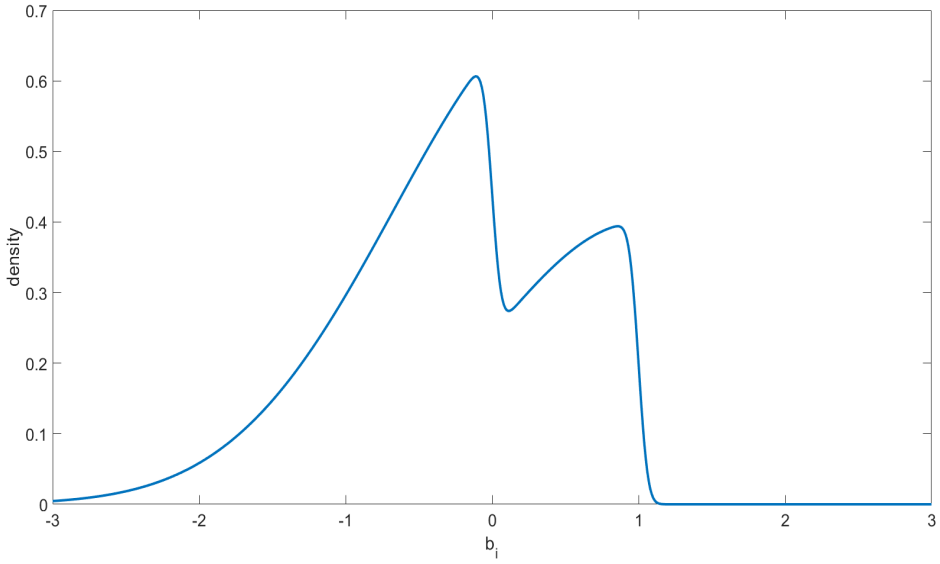


Figure 15: The plots of the average lower bound values over the last 500 steps for the CMGVA and mixture of normals variational approximation for the Bayesian logistic regression model for the ionosphere and krkp datasets

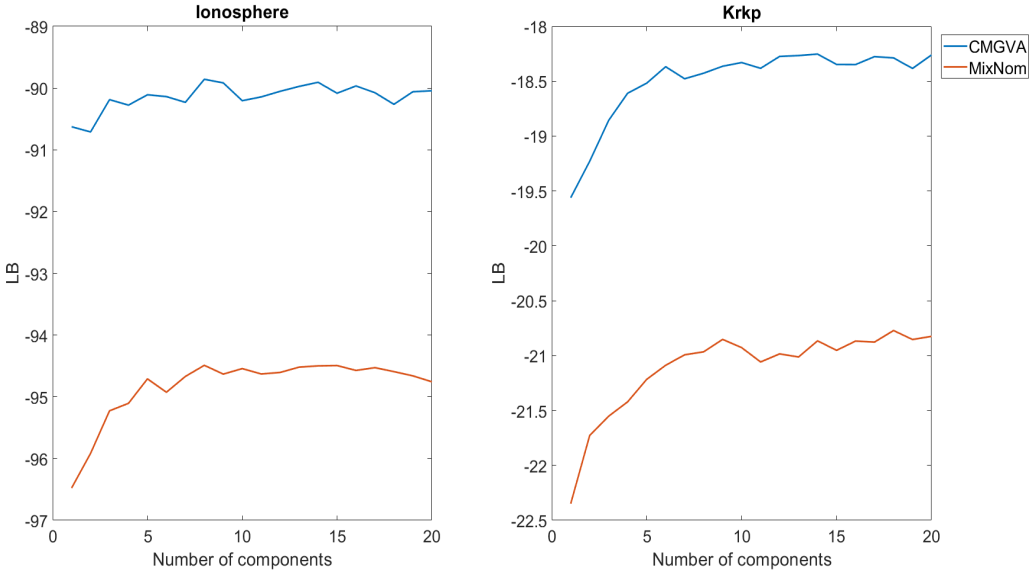


Figure 16: The plots of the average lower bound values over the last 500 steps for the CMGVA and mixture of normals variational approximation for the Bayesian logistic regression model for the mushroom and spam datasets

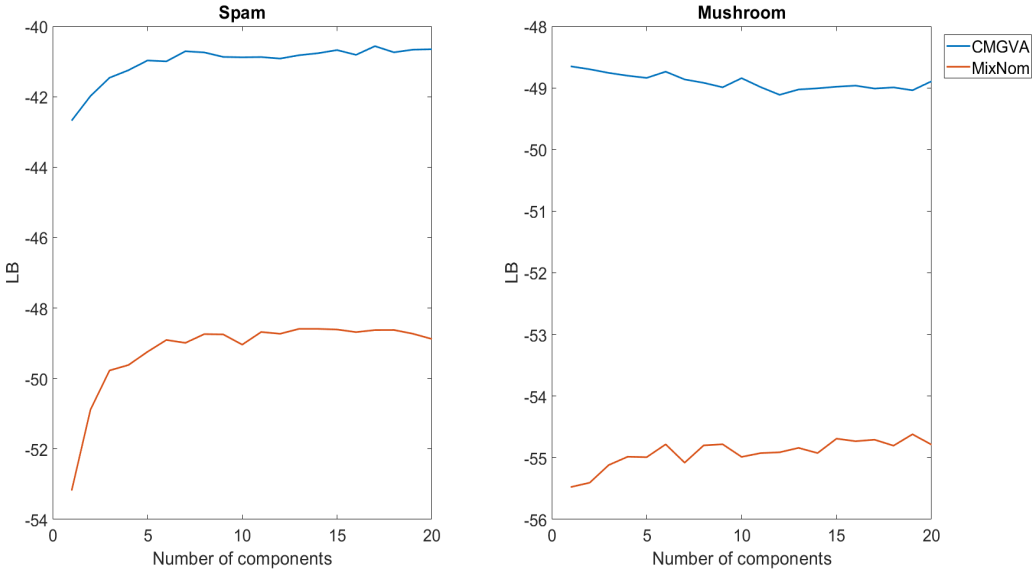
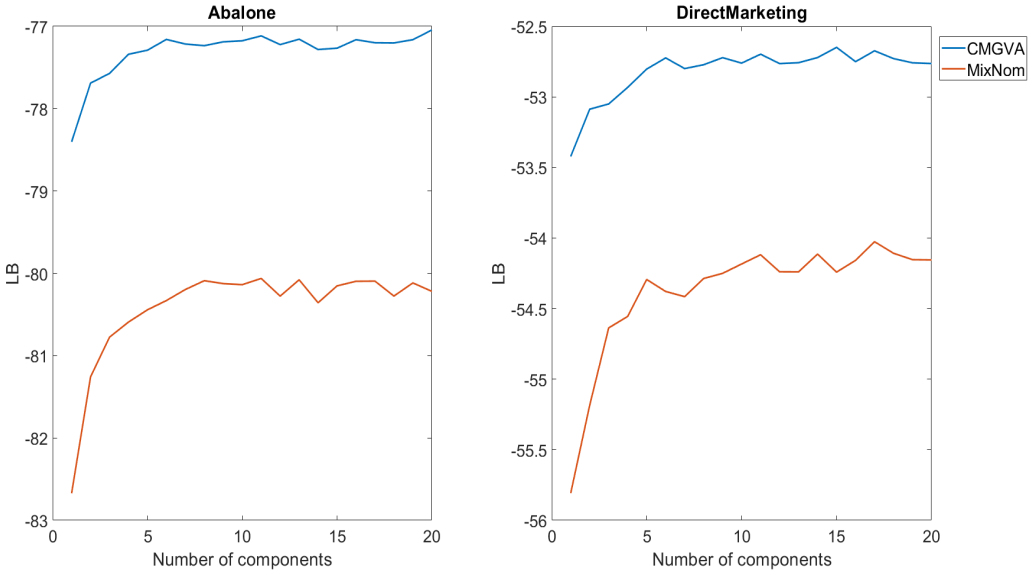


Figure 17: The plots of the average lower bound values over the last 500 steps for the CMGVA and mixture of normals variational approximation for the Bayesian linear regression model for the abalone and direct marketing datasets



One example where the complex prior distributions may arise is in the context of variable selection for Bayesian linear and logistic regression models. Suppose that some parameters are assumed to be positive or zero. Then, we can take the prior distribution for each regression parameter to be mixture of gammas with two components; the first component is centred very close to zero and has small variance and the second one is centred away from zero with much larger variance.

5.4 Flexible Bayesian Regression with a Deep Neural Network

Deep feedforward neural network (DFNN) models with binary and continuous response variables are widely used for classification and regression in the machine learning literature. The DFNN method can be viewed as a way to efficiently transform a vector of p raw covariates $X = (X_1, ..., X_p)^\top$ into a new vector Z having the form

$$Z := f_L\left(W_{L,} f_{L-1}\left(W_{L-1}, ..., f_1\left(W_1, X\right)\right)\right). \tag{18}$$

Each $Z_l = f_l\left(W_l, Z_{l-1}\right)$, $l = 1, ..., L$, is called a hidden layer, L is the number of hidden layers in the network, $W = (W_1, ..., W_L)$ is the set of weights and $Z_0 = X$ by construction. The function $f_l\left(W_l, Z_{l-1}\right)$ is assumed to be of the form $h_l\left(W_l Z_{l-1}\right)$, where W_l is a matrix of weights that connect layer l to layer $l+1$ and $h_l\left(\cdot\right)$ is a scalar activation function. Estimation in complex high dimensional models like DFNN re-

gression models is challenging. This section studies the accuracy of posterior densities and the predictive performance of the model based on CMGVA and compares it to the Gaussian copula and the mixture of normals for a DFNN regression model with continuous responses; see Goodfellow et al. (2016) for a comprehensive recent discussion of DFNNs and other types of neural networks.

Consider a dataset D with n observations, with y_i the scalar response and $x_i = (x_{i1}, ..., x_{ip})^\top$ the vector of p covariates. We consider a neural network structure with the input vector x and a scalar output. Denote $z_l := f_l(x, w)$, $l = 1, ..., M$, the units in the last hidden layer, w is the vector of weights up to the last hidden layer, and $b = (b_0, b_1, ..., b_M)^\top$ are the weights that connect the variable z_l, $l = 1, ..., M$, to the output y. The model, with a continuous response y, can be written as

$$\begin{aligned}
\sigma^2 &\sim Gamma(1, 10), && (19)\\
\tau^2 &\sim Gamma(1, 10), && (20)\\
w_i &\sim SN\left(0, 1/\sigma^2, \alpha\right), \text{for } i = 1, ..., M_w, && (21)\\
b_l &\sim SN\left(0, 1/\sigma^2, \alpha\right), \text{for } l = 1, ..., M, && (22)\\
y|x, w, b, \tau &\sim N\left(b^\top z, 1/\tau^2\right), && (23)
\end{aligned}$$

where M_w is the number of weight parameters; $SN(B; 0, 1/\sigma^2, \alpha)$ is a skew-normal density distribution of Azzalini (1985) with density $2\sigma\phi(\sigma B)\Phi(\alpha\sigma B)$, and $Gamma(1, 10)$ is the gamma distribution with shape parameter 1 and scale parameter 10. The hyperparameter α is set to 4; Miller et al. (2017) uses similar priors for σ^2 and τ^2.

All the examples use the rectified linear unit (ReLU) $\max(0, x) := h(x)$ as an activation function, unless otherwise stated; ReLU is widely aqpplied in the deep learning literature (Goodfellow et al., 2016) because it is easy to use within optimization as it is quite similar to a linear function, except that it outputs zero for negative values of x.

To evaluate the prediction accuracy of a DFNN regression model estimated by the Gaussian copula variational approximation, the CMGVA, and the mixture of normals, we consider the partial predictive score (PPS) defined as

$$\text{PPS} := -\frac{1}{n_{test}} \sum_{\text{test data}} \log p\left(y_i|x_i, \widehat{\theta}\right),$$

with $\widehat{\theta}$ a posterior mean estimate of the model parameters. The lower the PPS value, the better the prediction accuracy. In this example: the number of factors is set to 1 for all $k = 1, ..., 20$ components in the mixture; $S = 100$ samples are used to estimate

the gradients of the lower bound; the training of the neural net is stopped if the lower bound does not improve after 100 iterations; to reduce the noise in estimating the lower bound, we take the average of the lower bound over a moving window of 100 iterations (see Tran et al., 2017, for further details); the step sizes are set to the values given in Section 4.4, except α_μ is set to 0.001.

We consider the two datasets used in Section 5.3: the direct marketing dataset and the abalone dataset. The direct marketing dataset consists of 1000 observations, of which 900 are used for training the neural networks and the rest for testing. The abalone dataset has 4177 observations; we use 85% of the data for training and the rest for testing. Both datasets have continuous responses.

A neural net with a (11,5,5,1) structure is used for the direct marketing dataset, i.e. the input layer has 11 variables, there are two hidden layers each having 5 units and there is a (one) scalar output. The first layer has $11 \times 5 = 55$ w parameters, the second layer has $6 \times 5 = 30$ w parameters (including the intercept term), and 6 b parameters (including the intercept term); this gives a total of 91 parameters. A neural net (9,5,5,1) structure is used for the abalone data set, and similar calculation can be done for it to give a total of 75 parameters.

This section studies the accuracy of the posterior density estimates and the predictive performance based on CMGVA and compares it to the standard Gaussian copula and the mixture of normals variational approximations. Figure 18 shows the average lower bound value over the last 100 steps of the optimisation algorithm for the two datasets. The figure shows that estimating a neural network regression models with 3 to 4-components of the CMGVA increases the lower bound significantly for the direct marketing dataset compared to a Gaussian copula. The lower bound for the abalone dataset first decreases the lower bound up to 9-components and then increases it significantly up to 20-components for the CMGVA. The optimal number of components for the direct marketing and abalone datasets are 9 and 20, respectively; the optimal CMGVA is much better than Gaussian copula and mixture of normals variational approximation for both datasets. Figure 19 plots the average of PPS values over the last 100 steps of the optimisation algorithm for the two datasets–the lower the PPS values, the better the predictive performance. The figure shows that there are substantial predictive improvements by estimating $K = 3$ or 4 components of the mixture for the direct marketing dataset; there are no significant improvements after $K = 7$ components. For the abalone dataset, the PPS is signifcantly improved after $K = 10$ components; this is in line with the plot of the lower bound in Figure 18.

Figures 20 and 21 display the kernel density estimates of some of the marginal posteriors of the parameters of the deep neural network regression models from the direct

marketing and abalone datasets, respectively. The figures show that the CMGVA is able to approximate multimodal, skewed, and heavy-tailed posterior distributions of the model parameters and is different to the estimates from the Gaussian copula variational approximation. This confirms the usefulness of the CMGVA for complex and high-dimensional Bayesian deep neural network regression models.

Figure 18: The plots of the average lower bound values over the last 100 steps for the CMGVA and mixture of normals variational approximation for the DFNN regression model for the direct marketing and abalone datasets

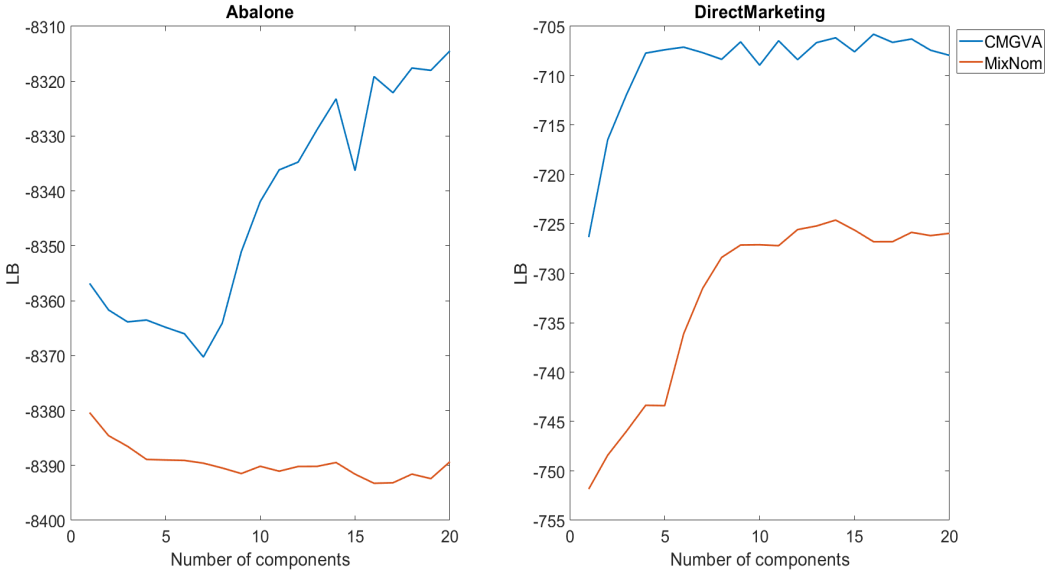


Figure 19: The plots of the average PPS values over the last 100 steps for the CMGVA and mixture of normals variational approximation for the DFNN regression model for the direct marketing and abalone datasets

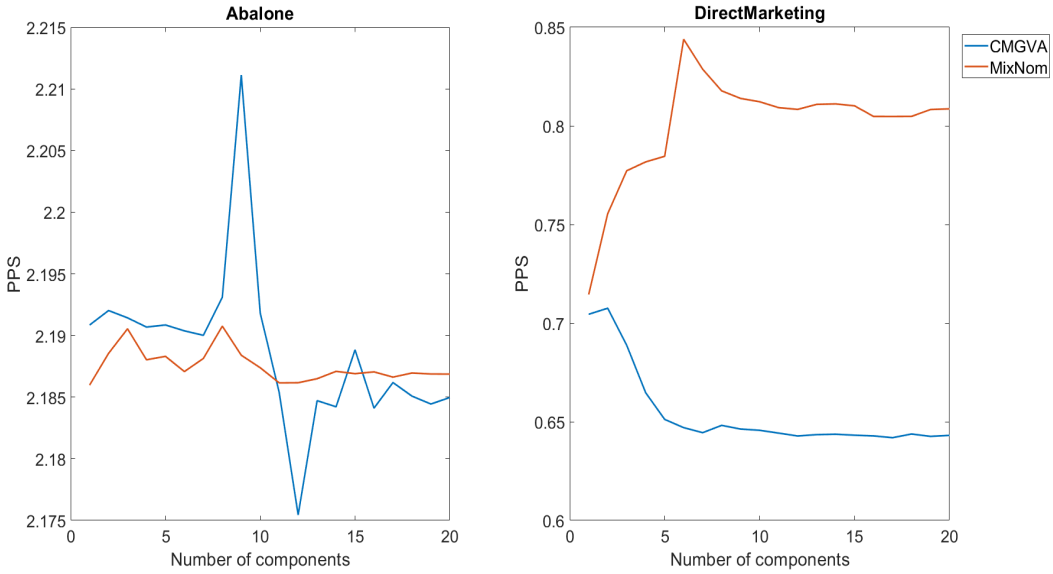


Figure 20: Kernel density estimates of some of the marginal parameters approximated with the Gaussian copula of Smith et al. (2020) and the CMGVA with 9 components for the direct marketing dataset. Marginal 11 means the 11th marginal posterior of the parameters (w, b).

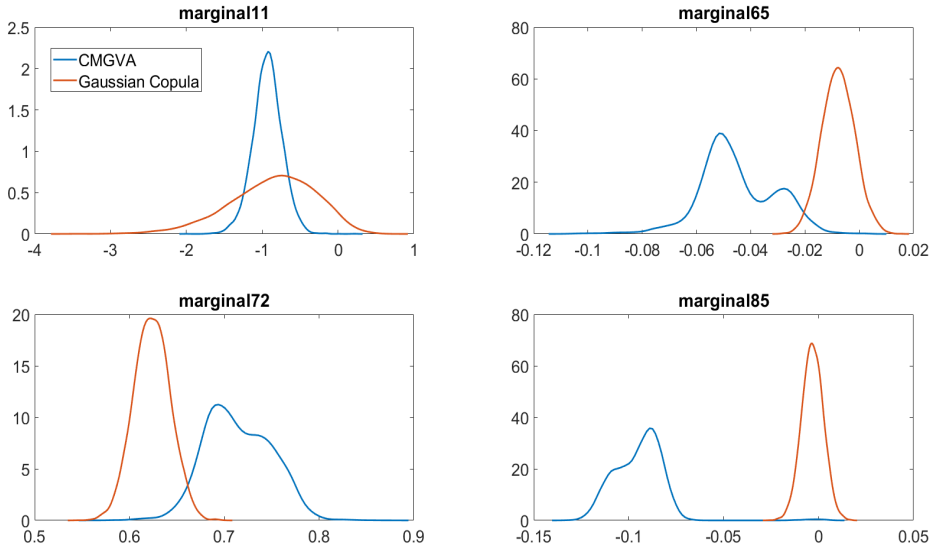
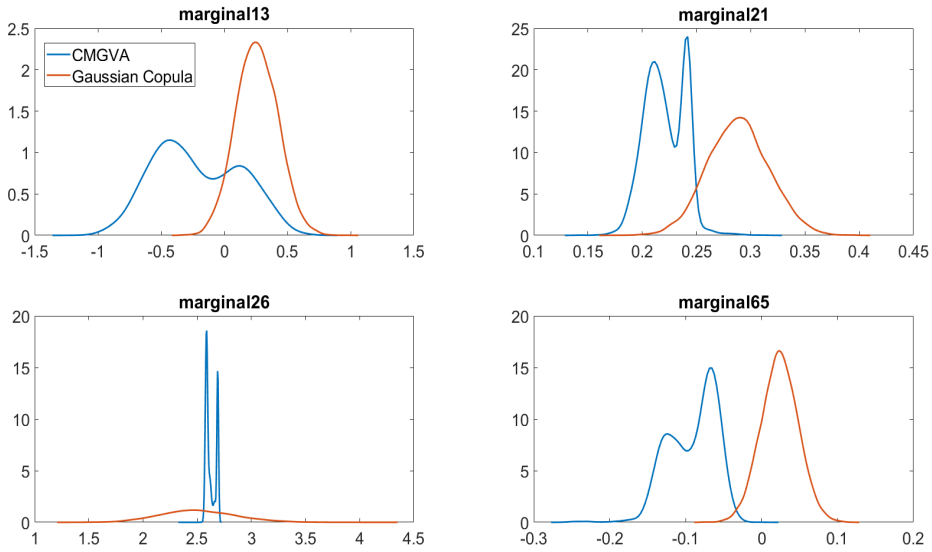


Figure 21: Kernel density Estimates of some of the marginal parameters approximated with Gaussian copula of Smith et al. (2020) and the CMGVA with 20 components for the abalone dataset. Marginal 11 means the 11th marginal posterior of the parameters (w, b)



6 Conclusions

The article approximates a posterior distribution by a copula of a mixture of normals (CMGVA), and constructs a variational method to estimate the approximation. The VB method is made efficient by using the natural gradient and control variates. Our approach of adding one component at a time provides a practical variational inference method that constructs an increasingly complicated posterior approximation and is an extension and refinement of state-of-the-art Gaussian copula variational approximation in Smith et al. (2020). The CMGVA approach applies to a wide range of Bayesian models; we apply it to four complex examples, including Bayesian deep learning regression models, and show that it improves upon the Gaussian copula and mixture of normals variational approximations in terms of both inference and prediction. Our article uses a factor structure for the covariance matrix in our variational approximation, but it is straightforward to extend the variational approach to consider other sparse forms of the covariance structure, such as a sparse Cholesky factorisation as in Tan et al. (2020).

7 Acknowledgement

The research of Robert Kohn was partially supported by an ARC Center of Excellence grant CE140100049.

References

Amari, S. (1998). Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276.

Archer, E., Park, I. M., Buesing, L., Cunningham, J., and Paninski, L. (2016). Black box variational inference for state space models. arXiv:1511.07367.

Azzalini, A. (1985). A class of distributions which includes the normal ones. *Scandinavian Journal of Statistics*, 12(2):171–178.

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

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT2010)*, pages 177–187. Springer.

Campbell, T. and Li, X. (2019). Universal boosting variational inference. *arXiv:1906.01235v2*.

Challis, E. and Barber, D. (2013). Gaussian Kullback-Leibler approximate inference. *Journal of Machine Learning Research*, 14:2239–2286.

Del Moral, P., Doucet, A., and Jasra, A. (2006). Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society, Series B*, 68:411–436.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Guo, F., Wang, X., Broderick, T., and Dunson, D. B. (2016). Boosting variational inference. *Advances in Approximate Bayesian Inference, NIPS workshop, ArXiv: 1611.05559v2*.

Han, S., Liao, X., Dunson, D., and Carin, L. (2016). Variational Gaussian copula inference. In Gretton, A. and Robert, C. C., editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 829–838, Cadiz, Spain. PMLR.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347.

Jank, W. (2011). *Business Analytics for Managers*. Springer-Verlag, New York.

Jerfel, G., Wang, S. L., Fannjiang, C., Heller, K. A., Ma, Y., and Jordan, M. (2021). Variational refinement for importance sampling using the forward Kullback-Leibler divergence. In de Campos, C. and Maathuis, M., editors, *Uncertainty in Artificial Intelligence (UAI), Proceedings of the Thirty-Seventh Conference*.

Khan, M. E. and Lin, W. (2017). Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In Singh, A. and Zhu, X. J., editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54, pages 878–887. PMLR.

Khan, M. E. and Nielsen, D. (2018). Fast yet simple natural-gradient descent for variational inference in complex models. In *International Symposium on Information Theory and Its Applications, ISITA 2018, Singapore, October 28-31, 2018*, pages 31–35. IEEE.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning*

Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Kleijnen, J. P. C. and Rubinstein, R. Y. (1996). Optimisation and sensitivity analysis of computer simulation models by the score function method. *European Journal of Operational Research*, 88(3):413–427.

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., and Blei, D. M. (2017). Automatic differentiation variational inference. *Journal of machine learning research*, 18(14):1–45.

Lichman, M. (2013). UCI machine learning repository. *University of California, Irvine, School of Information and Computer Sciences*.

Lin, W., Khan, M. E., and Schmidt, M. (2019). Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3992–4002. PMLR.

Locatello, F., Khanna, R., Ghosh, J., and Ratsch, G. (2018). Boosting variational inference: an optimization perspective. In Storkey, A. and Perez-Cruz, F., editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 464–472. PMLR.

Miller, A. C., Foti, N. J., and Adams, R. P. (2017). Variational boosting: Iteratively refining posterior approximations. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2420–2429. PMLR.

Nott, D. J., Tan, S., Villani, M., and Kohn, R. (2012). Regression density estimation with variational methods and stochastic approximation. *Journal of Computational and Graphical Statistics*, 21:797–820.

Ong, M. H. V., Nott, D. J., and Smith, M. S. (2018). Gaussian variational approximation with a factor covariance structure. *Journal of Computational and Graphical Statistics*, 27(3):465–478.

Ormerod, J. T. and Wand, M. P. (2010). Explaining variational approximations. *American Statistician*, 64:140–153.

Paisley, J., Blei, D. M., and Jordan, M. I. (2012). Variational Bayesian inference with stochastic search. In *Proceedings of the 29th International Conference on Machine Learning*, ICML'12, pages 1363–1370, Madison, WI, USA. Omnipress.

Ranganath, R., Gerrish, S., and Blei, D. M. (2014). Black box variational inference. In Kaski, S. and Corander, J., editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 814–822, Reykjavik, Iceland. PMLR.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In Xing, E. P. and Jebara, T., editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Beijing, China. PMLR.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407.

Salimans, T. and Knowles, D. A. (2013). Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):741–908.

Smith, M., Maya, R. L., and Nott, D. J. (2020). High-dimensional copula variational approximation through transformation. *Journal of Computational and Graphical Statistics*.

Tan, L., Bhaksaran, A., and Nott, D. (2020). Conditionally structured variational Gaussian approximation with importance weights. *Statistics and Computing*, 30:1225–1272.

Tan, L. S. L. and Nott, D. J. (2018). Gaussian variational approximation with sparse precision matrices. *Statistics and Computing*, 28(2):259–275.

Titsias, M. and Lázaro-Gredilla, M. (2014). Doubly stochastic variational Bayes for non-conjugate inference. In Xing, E. P. and Jebara, T., editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1971–1979, Beijing, China. PMLR.

Tran, M. N., Nguyen, N., Nott, D., and Kohn, R. (2019). Bayesian deep net GLM and GLMM. *Journal of Computational and Graphical Statistics*, 29(1):97–113.

Tran, M. N., Nott, D., and Kohn, R. (2017). Variational Bayes with intractable likelihood. *Journal of Computational and Graphical Statistics*, 26(4):873–882.

Tukey, T. W. (1977). Modern techniques in data analysis. *NSP-sponsored regional research conference at Southeastern Massachesetts University, North Dartmount, Massachesetts*.

Yeo, I. K. and Johnson, R. A. (2000). A new family of power transformations to improve normality or symmetry. *Biometrika*, 87(4):954–959.