Contents

1 Bayesian Inference

1.1 Bayes' Rule

In Bayesian statistics, we are interested in making inferences about an unknown parameter θ using what is known as the posterior distribution, $p(\theta|y)$, where y is our observed data. Bayes' rule provides a form for this distribution,

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)}, \tag{1.1}$$

which shows that the posterior can be found by taking the product of the likelihood, $p(y|\theta)$ and the prior distribution, $p(\theta)$, and normalising with a constant $p(y)$. In practice $p(y)$ is unknown and can be found from

$$p(y) = \int_\theta p(y|\theta)p(\theta)d\theta. \tag{1.2}$$

Finding the value of $p(y)$ is often the most difficult part of working with Bayesian statistics. In some simple models, this integration can be solved analytically, but for most models of interest it is intractable and we can only work with the un-normalised version

$$p(\theta|y)^* = p(y|\theta)p(\theta). \tag{1.3}$$

This work will concern itself with two of the methods used in the case where (**??**) is intractable, exact methods such as MCMC, which will be discussed later in this section as well as an approximation method known as Variational Bayes which appears in section two.

1.2 Exact Bayesian Computation

1. Numerical Integration

It might be tempting to numerically approximate (**??**) by evaluting the integrand over a grid and applying methods such as the Trapezoid Rule or Adaptive Quadrature. These may be suitable for integration in a low dimension, but if we have a grid with G points in one dimension, then a two dimensional integration requires G^2 points, and a three dimensional integration requires G^3 points. In general, the computational time increases exponentially with the dimension of θ in numerical integration. In almost all cases of interest, θ is a high-dimensional vector which makes numerical integration prohibitively expensive. This is known as the curse of dimensionality, where the difficulty of the problem increases quickly with the dimension of θ and requires us to use more sophisticated methods.

2. Importance Sampling

Candidate Density Accept Reject Good candidate is hard but useful VB as a candidate

3. Gibbs Sampling

Breaks down into simple problems Becomes linear instead of exponential Exact sampling

4. Metropolis Hastings within Gibbs

Gibbs only works exactly on some problems Approximation within Gibbs Also can benefit from VB

2 Variational Bayes

2.1 Estimation as an optimisation problem

In the event that our dataset is too large, or our model is too complex, for MCMC to converge within a reasonable time frame we must resort to approximation. One important question that might be asked then is *'How do I measure how good my approximation is?'*. If two distributions come from the same family, one might consider the difference in the parameters, but it is trivial to come up with distributions with slight parameter differences and vastly different regions of probability mass. Consider the difference between a $\mathcal{N}(0, 0.1)$ and a $\mathcal{N}(0.5, 0.1)$ distribution, the change in mean is only 0.5 but there is little overlap in the densities. Further, large values for the variance allow large changes in mean with very little change in the resulting densities. If the distributions come from different families it is almost impossible to use parameters to measure the degree of closeness between two distributions. Instead we must move from the parameter space to the distribution space, where one measure of the difference between distributions $p(\theta)$ and $q(\theta)$ is the Kullback-Leibler (KL) divergence (**?**), defined by

$$KL[q(\theta)||p(\theta)] = \int q(\theta) \ln \left(\frac{q(\theta)}{p(\theta)} \right) d\theta. \tag{2.1}$$

KL divergence is a non-negative, assymetric measure of the difference between $p(\theta)$ and $q(\theta)$ that will equal zero if and only if $p(\theta) = q(\theta)$ almost everywhere. It has origins in information theory, and can be interpreted as: *'Given that I know $p(\theta)$ for some $\theta \in \Theta$, how much extra information is required, on average, to know the value of $q(\theta)$?'* There are examples of the literature of approximations with other measures of divergence, such as **?** introducting Expectation Propogation, which minimises the reverse measure $KL[p(\theta)||q(\theta)]$,

which was extended to Power-EP in **?**, which aims to minimisea the more general $\alpha-$ divergence (**?**).

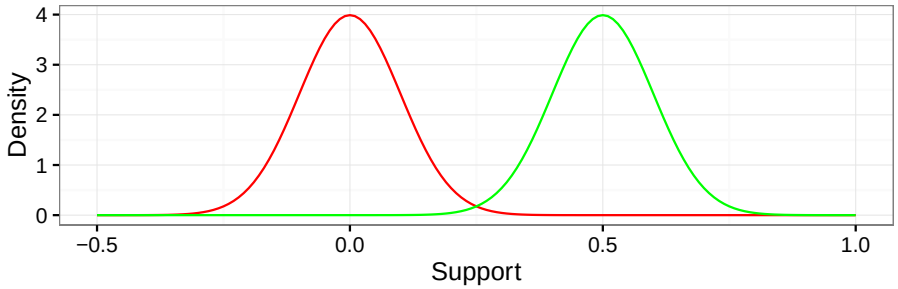

Figure 1: Red: A Normal(0, 0.1) distribution. Green: A Normal(0.5, 0.1) distribution.

Variational Bayes aims to find the approximating distribution, $q(\theta|\lambda)$, that minimises $KL[q(\theta|\lambda)||p(\theta|y)]$. This approximating distribution is parameterised by a λ vector, each element of which may be a function of y, the hyperparameters of the prior distribution, and even of other elements in λ. The optimal approximation, in the sense of having the minimum KL divergence with the posterior, can be derived by beginning with the unknown constant, $\ln(p(y))$.

$$\begin{aligned}
\ln(p(y)) &= \int_\theta q(\theta|\lambda)\ln(p(y))d\theta \\
&= \int_\theta q(\theta|\lambda)\ln\left(\frac{p(y,\theta)}{p(\theta|y)}\right)d\theta \\
&= \int_\theta q(\theta|\lambda)\ln\left(\frac{p(y,\theta)}{p(\theta|y)}\frac{q(\theta|\lambda)}{q(\theta|\lambda)}\right)d\theta \\
&= \int_\theta q(\theta|\lambda)\ln\left(\frac{q(\theta|\lambda)}{p(\theta|y)}\right) + \ln\left(\frac{p(y,\theta)}{q(\theta|\lambda)}\right)d\theta \\
&= \int_\theta q(\theta|\lambda)\ln\left(\frac{q(\theta|\lambda)}{p(\theta|y)}\right)d\theta + \int_\theta q(\theta|\lambda)\ln\left(\frac{p(y,\theta)}{q(\theta|\lambda)}\right)d\theta \\
&= KL[q(\theta||\lambda)||p(\theta|y)] + \mathcal{L}(q(\theta|\lambda), y)
\end{aligned}$$

where

$$\begin{aligned}
\mathcal{L}(q(\theta|\lambda), y) &= \int_\theta q(\theta|\lambda)\ln\left(\frac{p(y,\theta|\lambda)}{q(\theta|\lambda)}\right)d\theta \\
&= \int_\theta q(\theta|\lambda)\ln(p(y,\theta|\lambda))d\theta - \int_\theta q(\theta|\lambda)\ln(q(\theta|\lambda))d\theta. \qquad (2.2)
\end{aligned}$$

Hence we have that

$$KL[q(\theta|\lambda)||p(\theta|y)] = \ln(p(y)) - \mathcal{L}(q(\theta|\lambda), y). \qquad (2.3)$$

From (**??**) it is clear that by maximising $\mathcal{L}(q(\theta|\lambda), y)$ with respect to $q(\theta|\lambda)$ the KL divergence between the unknown true posterior $p(\theta|y)$ and the approximating distribution $q(\theta|\lambda)$ will be minimised. Hence the Variational Bayes solution will be to maximise the Evidence Lower Bound (ELBO), $\mathcal{L}(q(\theta|\lambda), y)$, in (**??**).

2.2 Mean Field Variational Bayes

Mean Field Variational Bayes (MFVB) has origins in the physics literature (**?**) and uses the assumption that the approximation dististribution factorises,

$$q(\theta|\lambda) = \prod_i q_i(\theta_i|\lambda_i). \qquad (2.4)$$

This assumption is known as the Mean Field assumption and is has been widely used as it greatly simplifies maximisation of the ELBO (**?**; **?**). Each

component θ_i may be a scalar or a vector, and λ_i is the component of the λ vector that parameterises the relevant factor $q_i(\theta_i|\lambda_i)$. From here, we will use the shorthand notation that $q_i = q_i(\theta_i|\lambda_i)$ and $q_{\setminus i} = \prod_{j \neq i} q_j$. Maximising the ELBO with respect to q_i is analytically involved but computationally simple, as (**??**) can be expressed as a function of q_i only and then maximised with respect to each q_i individually.

$$\begin{aligned}
\mathcal{L}(q_i) &= \int_\theta q_i q_{\setminus i} \ln(p(y,\theta)) d\theta - \int_\theta q_i q_{\setminus i} \ln(q_i q_{\setminus i}) d\theta \\
&= \int_{\theta_i} \int_{\theta_{\setminus i}} q_i q_{\setminus i} (\ln(p(y,\theta)) - \ln(q_i)) d\theta_i d\theta_{\setminus i} - \int_{\theta_i} \int_{\theta_{\setminus i}} q_i q_{\setminus i} \ln(q_{\setminus i}) d\theta_i d\theta_{\setminus i} \\
&= \int_{\theta_i} q_i \left(\int_{\theta_{\setminus i}} q_{\setminus i} \ln(p(y,\theta)) d\theta_{\setminus i} - \ln(q_i) \right) d\theta_i \\
&- \int_{\theta_{\setminus i}} q_{\setminus i} \ln(q_{\setminus i}) d\theta_{\setminus i} \qquad (2.5) \\
&= \int_{\theta_1} q_i \ln \left(\frac{\exp(\mathbb{E}_{q_{\setminus i}}[\ln(p(y,\theta))])}{q_i} \right) d\theta_i + c \\
&= -KL(q_i || \exp(\mathbb{E}_{q_{\setminus i}}[\ln(p(y,\theta))])) + c \qquad (2.6)
\end{aligned}$$

as the rightmost component of (**??**) is constant with respect to q_i.

We can see that by holding $q_{\setminus_i}$ fixed, we can maximise the ELBO with respect to q_i by minimising the KL Divergence between q_i and $\exp(\mathbb{E}_{q_{\setminus i}}[\ln(p(y,\theta))])$ in (**??**). This is easily done by setting

$$q(\theta_i|\lambda_i) \propto \exp(\mathbb{E}_{q_{\setminus i}}[\ln(p(y,\theta))]). \qquad (2.7)$$

If the likelihood and prior for θ_i form an exponential family conjugate pair, then this maximisation proceeds by keeping the distributional family constant and simply updating the parameters λ_i. Otherwise, we can find the optimal distributional family for any q_i by taking the log-joint density and then ignoring any term that doesn't depend on θ_i. Next, take the expectation of any remaining $\theta_{j \neq i}$ and then match to the kernel of a known distribution. If it does not match a known distribution, we may need to make a further approximation $q_i(\tilde{\theta_i}|\lambda_i)$ which has a recognizable distribution. One method as used in **?** uses a Laplace approximation for the otherwise unrecognizable $q_i(\theta_i|\lambda_i)$ with reasonable results.

These steps are almost the same as those used to find a conditional distribution $p(\theta_i|\theta_{j \neq i}, y)$, with dependence on other parameters replaced by their expectations. Hence, the optimal approximating family for θ_i, $q(\theta_i|\lambda_i)$, will come from the same distributional family as the conditional distribution $p(\theta_i|\theta_{j \neq i}, y)$, if it exists in a recognisable form such as the exponential family. Deriving the updating algorithm for MFVB takes more analytical effort than finding the conditionals for a Gibbs MCMC scheme, as we must then find the values of each λ_i implied by (**??**), which are often a function of $\lambda_{j \neq i}$ through the expectations.

Given these optimal distributional families, an algorithm that cycles through all of the λ_i and sets each to its maximising value is known as a coordiante ascent algorithm. The algorithm must continuously iterate between each parameter until $\mathcal{L}(q(\theta|\lambda), y)$ converges with some pre-defined threshold noting that the λ may depend on each other. If λ has dimension k, the algorithm follows below.

Input: Log Joint Density
Result: Mean Field Approximation
Initialise λ randomly;
while Not converged **do**
  for $i = 1$ **to** k **do**
    Hold $\lambda_{j \neq i}$ fixed;
    Use (**??**) to match $q(\theta_i|\lambda_i)$ to a known distribution;
    Update λ_i using the most recent values of $\lambda_{j \neq i}$;
  end
end

Algorithm 1: Coordinate Ascent for MFVB

2.3 Stochastic Gradient Ascent

In many cases, the posterior is too complex to be captured by a factorising approximation and we desire our distribution $q(\theta|\lambda)$ to capture dependence between parameters. In this case, the easy maximisation in MFVB is unavailable, and we must resort to what is called Stochastic Variational Bayes (SVB) using a gradient ascent algorithm developed by **?** and **?**.

To maximise the function $\mathcal{L}(q(\theta|\lambda), y)$ we can take the derivative of $\mathcal{L}(q(\theta|\lambda), y)$ with respect to λ and use the updating step:

$$\lambda^{(m)} = \lambda^{(m-1)} + \rho^{(m)} \nabla_\lambda \mathcal{L}(q(\theta|\lambda^{(m-1)}), y), \tag{2.8}$$

where $\nabla_\lambda \mathcal{L}(q(\theta|\lambda^{(m-1)}), y)$ is the vector of partial derivatives of $\mathcal{L}(q(\theta|\lambda^{(m-1)}), y)$ with respect to each element of λ. This update requires initial values $\lambda^{(0)}$ and some sequence $\rho^{(m)}$. If $\rho^{(m)}$ is chosen to satisfy the following conditions, it is a Robbins-Monro sequence and the algorithm is guaranteed to converge to a local maximum.

$$\lim_{m \to \infty} \rho^{(m)} = 0$$
$$\sum_{m=1}^{\infty} \rho^{(m)} = \infty$$
$$\sum_{m=1}^{\infty} (\rho^{(m)})^2 < \infty.$$

Whilst a global maximum is desired, the ELBO can have a problem specific shape that makes finding the global maximum extremely difficult, as we do not

know how many stationary points exist. The dimension of the ELBO is the dimension of the λ vector, which is often much greater than the dimension of the parameters θ so a grid search is suspect to the curse of dimensionality. To alleiviate this problem, we can start the algorithm at a range of initial values choose the converged value with the highest value of the ELBO.

SVB does not provide the family of the the optimal approximating distribution, unlike under the mean field assumption. To run SVB we may need to try many approximating distributions and then choose the $q(\theta|\lambda)$ that has the highest ELBO, and hence lowest KL divergence to the true posterior. We must restrict the approximation to distributions that satisfy the condition that the order of differentation of the ELBO with respect to λ and integration with respect to θ are interchangable. In this case, we can calculate derivatives of the ELBO.

$$\begin{aligned}
\nabla_\lambda \mathcal{L}(q(\theta|\lambda), y) &= \nabla_\lambda \int_\theta q(\theta|\lambda) \left(\ln(p(y,\theta)) - \ln(q(\theta|\lambda))\right) d\theta \\
&= \int_\theta \nabla_\lambda [q(\theta|\lambda) \left(\ln(p(y,\theta)) - \ln(q(\theta|\lambda))\right)] d\theta \qquad (2.9) \\
&= \int_\theta q(\theta|\lambda) \nabla_\lambda [(\ln(p(y,\theta)) - \ln(q(\theta|\lambda))] d\theta \\
&+ \int_\theta \nabla_\lambda [q(\theta|\lambda)] \left(\ln(p(y,\theta)) - \ln(q(\theta|\lambda))\right) d\theta \qquad (2.10)
\end{aligned}$$

Note that the first integral on the right hand side of (**??**) equals 0 as

$$\int_\theta q(\theta|\lambda) \nabla_\lambda [(\ln(p(y,\theta))] d\theta = 0 \qquad (2.11)$$

and

$$\begin{aligned}
\int_\theta -q(\theta|\lambda) \nabla_\lambda [\ln(q(\theta|\lambda))] d\theta &= -\int_\theta \frac{\nabla_\lambda q(\theta|\lambda)}{q(\theta|\lambda)} q(\theta|\lambda) d\theta \\
&= -\nabla_\lambda \int_\theta q(\theta|\lambda) d\theta \\
&= 0
\end{aligned}$$

Then we continue from (**??**) with

$$\begin{aligned}
\nabla_\lambda \mathcal{L}(q(\theta|\lambda), y) &= \int_\theta \nabla_\lambda [q(\theta|\lambda)] \left(\ln(p(y,\theta)) - \ln(q(\theta|\lambda))\right) d\theta \\
&= \int_\theta q(\theta|\lambda) \nabla_\lambda [\ln(q(\theta|\lambda))] \left(\ln(p(y,\theta)) - \ln(q(\theta|\lambda))\right) d\theta \\
&= E_q \left(\nabla_\lambda [\ln(q(\theta|\lambda))] \left(\ln(p(y,\theta)) - \ln(q(\theta|\lambda))\right)\right). \qquad (2.12)
\end{aligned}$$

We can compute a Monte-Carlo estimate of (**??**) by

$$\nabla_\lambda \mathcal{L}(q(\theta|\lambda)) \approx \frac{1}{S}\sum_{s=1}^{S} \nabla_\lambda[\ln(q(\theta_s|\lambda))](\ln(p(y,\theta_s)) - \ln(q(\theta_s|\lambda))) \tag{2.13}$$

where θ_s, $s = 1, \dots S$, are simulated from the most recent iteration of $q(\theta|\lambda)$.

As the distribution $q(\theta|\lambda)$ is specified by the user, the main restriction on the use of SVB is that the log-joint density $\ln(p(y,\theta))$ is able to be evaluated. Running an SVB algorithm involves very little analytical effort unlike MFVB, most of the difficulty is calculating the score function of the approximating distribution, $\nabla_\lambda[\ln(q(\theta_s|\lambda))]$. But as this is part of the approximation and not the true model the code to run this could potentially be re-used in many different models.

Input: Log Joint Density, Approximation family q
Result: Variational Approximation
Initialise λ;
while Not converged **do**
  Simulate θ^s for $s = 1, \dots S$ from $q(\theta|\lambda^{(m-1)})$;
  for $i = 1$ **to** k **do**
    Calculate $\nabla_{\lambda_i} =$
    $1/S \sum_{s=1}^{S} \nabla_{\lambda_i}[\log(q(\theta^s|\lambda^{(m-1)})](\log(p(\theta^s, y)) - \log(q(\theta^s|\lambda^{(m-1)})))$;
  end
  Set $\lambda^{(m)} = \lambda^{(m-1)} + \rho^{(m)}\nabla_\lambda$;
  Set $m = m + 1$;
end

Algorithm 2: Gradient Ascent for SVB

2.4 SVB Extensions

1. AdaGrad

If the gradient is small, we want the learning rate, ρ_t, to be large, and if the gradient is large, we want the learning rate to be small. AdaGrad (**?**) is a method of setting the learning rate to dynamically adapt as the values of the calculated gradients changes. Further, AdaGrad allows the step size of each element of λ to move independently of the other elements of λ.

If $\nabla_\lambda^{(m)}$ is the vector of gradients at iteration m, define

$$G^{(m)} = \sum_{i=1}^{m} \nabla_\lambda^i (\nabla_\lambda^i)^T, \tag{2.14}$$

the sum of the outer products of the gradients up to iteration m. Then set

$$\rho^{(m)} = \eta \text{diag}(G^{(m)})^{-1/2} \tag{2.15}$$

with some preset tuning parameter η. If we have k elements in λ, $G^{(m)}$ is a $k \times k$ matrix so we have one element in $\rho^{(m)}$ for each element of λ. As we only need the diagonal of $G^{(m)}$, this may be calculated in linear time.

2. Reparameterisation

The reparameterisation trick was popularised following **?** and allows the gradient of the ELBO to be simplified.

Consider a parameter free auxillary distribution $q(\epsilon)$ and differentiable transformation $f(\cdot, \cdot)$ such that $\theta = f(\epsilon, \lambda)$. Examples include a location-scale transformation from a standard normal or an inverse-CDF transform from a uniform$(0, 1)$ variable. Note that

$$q_\theta(\theta|\lambda) = q_\epsilon(\epsilon) \left| \frac{d\epsilon}{d\theta} \right|$$

where the parameters that govern the transform f are the same λ parameters as in $q(\theta|\lambda)$. Now (**??**) becomes

$$\begin{aligned} \nabla_\lambda \mathcal{L}(q(\theta|\lambda), y) &= \int_\epsilon \nabla_\lambda [q(\epsilon) \left(\ln(p(y, f(\epsilon, \lambda))) - \ln(q(f(\epsilon, \lambda)|\lambda))\right] d\epsilon \\ &= \int_\epsilon q(\epsilon) \nabla_\lambda \left[\ln(p(y, f(\epsilon, \lambda))) - \ln(q(f(\epsilon, \lambda)|\lambda)\right] d\epsilon \qquad (2.16) \\ &= \int_\epsilon q(\epsilon) \nabla_\lambda \left[\ln(p(y, f(\epsilon, \lambda))\right] d\epsilon \end{aligned}$$

as the second part on the right hand side of (**??**) is a score function with expectation equal to zero. This can be estimated using

$$\mathcal{L}(q(\theta|\lambda), y) \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_\lambda \left[\ln(p(y, f(\epsilon_s, \lambda)))\right]. \qquad (2.17)$$

with $\epsilon_s \sim q(\epsilon)$.

? explore how this reparameterised version can have orders of magnitude lower variance than the estimator in (**??**).

Input: Log Joint Density, Approximation family q
Result: Variational Approximation
Initialise λ;
while Not converged **do**
  Simulate ϵ^s for $s = 1, \dots S$;
  for $i = 1$ **to** k **do**
    Calculate $\nabla_{\lambda_i} = 1/S \sum_{s=1}^{S} \nabla_{\lambda_i} [\log(p(f(\epsilon^s, \lambda^{(m-1)}), y))]$
  end
  Set $\lambda^{(m)} = \lambda^{(m-1)} + \rho^{(m)} \nabla_\lambda^t$;
  Set $m = m + 1$;
end

Algorithm 3: Gradient Ascent for re-parameterised SVB

3. Control Variates

If reparameterisation is too difficult or otherwise unavailable, **?** introduce the use use of control variates to estimate the gradient with reduced variance.

Define

$$g(\theta, \lambda, y) = \nabla_\lambda[\ln(q(\theta_s|\lambda))](\ln(p(y, \theta_s)) - \ln(q(\theta_s|\lambda))), \tag{2.18}$$

the function estimated in (**??**), then for some other function h, define

$$\hat{g}(\theta, \lambda, y) = g(\theta, \lambda, y) - a(h - \mathbb{E}(h)). \tag{2.19}$$

Then we have that

$$\mathbb{E}(\hat{g}) = \mathbb{E}(g) \tag{2.20}$$

and

$$\text{Var}(\hat{g}) = \text{Var}(g) + a^2\text{Var}(h) - 2a\text{Cov}(g, h). \tag{2.21}$$

Solving the polynomial (**??**) in a shows that the variance of $\hat{g}$ is minimised by

$$\hat{a} = \text{Cov}(g, h)/\text{Var}(h). \tag{2.22}$$

Substituting (**??**) into (**??**) yields

$$\text{Var}(\hat{g}) = \text{Var}(g) - \text{Cov}(g, h)^2/\text{Var}(h). \tag{2.23}$$

We need to choose some function h that has a large covariance with g. **?** suggests that the an easy option is

$$h(\theta, \lambda) = \nabla_\lambda[\ln(q(\theta_s|\lambda))]. \tag{2.24}$$

As h is a score function, it has an expectation of zero and our estimator becomes

$$\nabla_\lambda\mathcal{L}(q(\theta|\lambda), y) \approx \frac{1}{S}\sum_{s=1}^{S} \nabla_\lambda[\ln(q(\theta_s|\lambda))](\ln(p(y, \theta_s)) - \ln(q(\theta_s|\lambda)) - \hat{a}), \tag{2.25}$$

where $\hat{a}$ is estimated from the sample variance and covariance of a subset of the S Monte-Carlo draws.

Input: Log Joint Density, Approximation family q
Result: Variational Approximation
Initialise λ;
while Not converged **do**
  Simulate θ_s for $s = 1, \dots S$ from $q(\theta|\lambda^{(m-1)})$;
  for $i = 1$ **to** k **do**
    Calculate $g_{\lambda_i} =$
    $1/S \sum_{s=1}^{S} \nabla_{\lambda_i}[\log(q(\theta^s|\lambda^{(m-1)})](\log(p(\theta^s, y)) - \log(q(\theta^s|\lambda^{(m-1)})))$;
    Calculate $h_{\lambda_i} = 1/S \sum_{s=1}^{S} \nabla_{\lambda_i}[\log(q(\theta^s|\lambda^{(m-1)}))]$;
    Use a subset of S to estimate $\hat{a} = \text{Cov}(h, g)/\text{Var}(g)$;
    Calculate $\nabla_{\lambda_i} = g_{\lambda_i} - \hat{a} h_{\lambda_i}$;
  end
  Set $\lambda^{(m)} = \lambda^{(m-1)} + \rho^{(m)} \nabla_{\lambda}$;
  Set $m = m + 1$;
end

Algorithm 4: Gradient Ascent for SVB with control variates

2.5 AR2 model example

Consider T observations from an AR(2) model conditioned on y_1 and y_2,

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \epsilon_t \text{ for } t = 3, \dots, T, \tag{2.26}$$

where ϵ are iid $\mathcal{N}(0, \sigma^2)$.
Using the independent priors:

$$\begin{aligned} p(\phi_1) &\sim \mathcal{N}(\bar{\phi_1}, \tau_1) \\ p(\phi_2) &\sim \mathcal{N}(\bar{\phi_2}, \tau_2) \\ p(\sigma^2) &\sim \mathcal{IG}(\text{shape} = a,\ \text{scale} = b), \end{aligned}$$

we have a model with zero stationary restrictions which can easily be sampled through a Gibbs MCMC scheme. This exact posterior can be used in comparision with the results of our variational approximations.

The Gibbs conditional distributions are derived in the appendix and are

$$\begin{aligned} p(\phi_1|\phi_2, \sigma^2, y_{1:T}) &\propto \mathcal{N}(\mu_1^G, \gamma_1^G) \\ p(\phi_2|\phi_1, \sigma^2, y_{1:T}) &\propto \mathcal{N}(\mu_2^G, \gamma_2^G) \\ p(\sigma^2|\phi_1, \phi_2, y_{1:T}) &\propto \mathcal{IG}(shape = \alpha^G, scale = \beta^G), \end{aligned}$$

where

$$
\begin{aligned}
\mu_1^G &= \frac{\tau_1(\sum_{t=3}^T y_t y_{t-1} - \phi_2 \sum_{t=3}^T y_{t-1} y_{t-2}) + \sigma^2 \bar{\phi_1}}{\tau_1 \sum_{t=3}^T y_{t-1}^2 + \sigma^2} \\
\gamma_1^G &= \frac{\sigma^2 \tau_1}{\tau_1 \sum_{t=3}^T y_{t-1}^2 + \sigma^2} \\
\mu_2^G &= \frac{\tau_2(\sum_{t=3}^T y_t y_{t-2} - \phi_1 \sum_{t=3}^T y_{t-1} y_{t-2}) + \sigma^2 \bar{\phi_2}}{\tau_2 \sum_{t=3}^T y_{t-2}^2 + \sigma^2} \\
\gamma_2^G &= \frac{\sigma^2 \tau_2}{\tau_2 \sum_{t=3}^T y_{t-2}^2 + \sigma^2} \\
\alpha^G &= a + T/2 - 1 \\
\beta^G &= b + 1/2 \sum_{t=3}^T \left((y_t - \phi_1 y_{t-1} - \phi_2 y_{t-2})^2 \right).
\end{aligned}
$$

100 data points are simulated from the model in (**??**) with true values of $\phi_1 = 0.8, \phi_2 = 0.15$ and $\sigma^2 = 2$. These values were chosen to induce a strong dependence between the ϕ parameters as it is close to the non-stationary region of $\phi_1 + \phi_2 > 1$. The resulting posterior is sampled with 100,000 draws from Gibbs MCMC, discarding the first 20% of all draws as a burn in. The resulting draws were strongly dependent so they were thinned by a factor of 20 to result in 4000 effectively independent draws. The total runtime for MCMC was 2.6 seconds. The dependence between ϕ_1 and ϕ_2 is apparent in figure (**??**), so a SGA algorithm that captures this dependence should be a better approximation than a MFVB algorithm that assumes independence.

The derivation of the MFVB approximation is also in the appendix and each marginal has the same form as the corresponding true conditional distribution above,

$$
\begin{aligned}
q(\phi_1|\lambda) &\propto \mathcal{N}(\mu_1^{MF}, \gamma_1^{MF}) \\
q(\phi_2|\lambda) &\propto \mathcal{N}(\mu_2^{MF}, \gamma_2^{MF}) \\
q(\sigma^2|\lambda) &\propto \mathcal{IG}(shape = \alpha^{MF}, scale = \beta^{MF}),
\end{aligned}
$$

where the parameterising vector $\lambda = (\mu_1^{MF}, \gamma_1^{MF}, \mu_2^{MF}, \gamma_2^{MF}, \alpha^{MF}, \beta^{MF})'$. The MVFB derivation yields the following updating equations for λ,

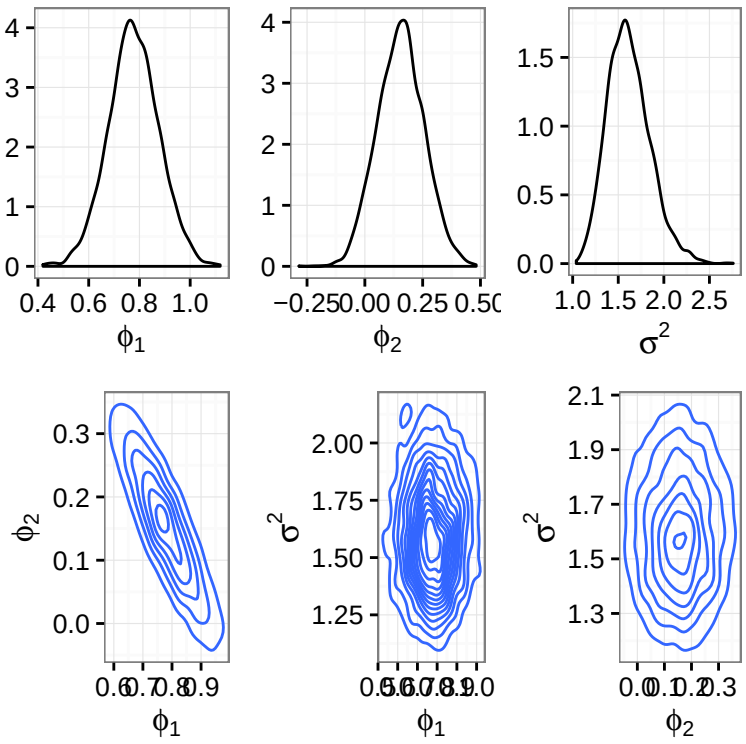

Figure 2: Marginal and bivariate density plots of the MCMC draws. 100,000 draws per parameter were sampled from (**??**) via Gibbs MCMC then reduced to 4000 each. Note the strong dependence between the two ϕ parameters.

$$
\begin{aligned}
\mu_1^{MF} &= \frac{\alpha^{MF}/\beta^{MF}(\sum_{t=3}^{T} y_t y_{t-1} - \mu_2 \sum_{t=3}^{T} y_{t-1} y_{t-2}) + \tau_1^{-1}\bar{\phi_1}}{\alpha^{MF}/\beta^{MF} \sum_{t=3}^{T} y_{t-1}^2 + \tau_1^{-1}} \\
\gamma_1^{MF} &= \left(\alpha^{MF}/\beta^{MF} \sum_{t=3}^{T} y_{t-1}^2 + \tau_1^{-1}\right)^{-1} \\
\mu_2^{MF} &= \frac{\alpha^{MF}/\beta^{MF}(\sum_{t=3}^{T} y_t y_{t-2} - \mu_1 \sum_{t=3}^{T} y_{t-1} y_{t-2}) + \tau_2^{-1}\bar{\phi_2}}{\alpha^{MF}/\beta^{MF} \sum_{t=3}^{T} y_{t-2}^2 + \tau_2^{-1}} \\
\gamma_1^{MF} &= \left(\alpha^{MF}/\beta^{MF} \sum_{t=3}^{T} y_{t-2}^2 + \tau_2^{-1}\right)^{-1} \\
\alpha^{MF} &= a + T/2 - 1 \\
\beta^{MF} &= b + 1/2 \sum_{t=3}^{T} \left((y_t^2 - ((\mu_1^{MF})^2 + \gamma_1^{MF}) y_{t-1}^2 - ((\mu_2^{MF})^2 + \gamma_2^{MF}) y_{t-2}^2\right) \\
&\quad + 2 \sum_{t=3}^{T} \left(\mu_1^{MF} \mu_2^{MF} y_{t-1} y_{t-2} - \mu_1^{MF} y_t y_{t-1} - \mu_2^{MF} y_t y_{t-2}\right).
\end{aligned}
$$

The forms of each component are similar to the conditionals derived for Gibbs sampling, with changes coming from linearising the expressions before taking expectations. Notice that all components of λ except α^{MF} depend on each other through the expectations in (**??**).

The SGA algorithm requires the specification of the approximating family so we make the following assumptions:

- The optimal marginal distribution for the dependent SGA approximation is the same as the independent MFVB approximation.
- There is no dependence between either ϕ parameter and σ^2.
- There is eliptical dependence between ϕ_1 and ϕ_2.

To determine the form of dependence between ϕ_1 and ϕ_2 we fit elliptical copulas to the MCMC draws and determined that the gaussian copula is the best fit by BIC. Gaussian marginals for each ϕ and a gaussian copula leads to ϕ_1 and ϕ_2 having a bivariate gaussian distribution.

Hence we have the following form for the SGA approximations,

$$
\begin{aligned}
q(\phi_1, \phi_2) &\sim \mathcal{N}(\mu, \Sigma) \\
q(\sigma^2) &\sim \mathcal{IG}(\alpha, \beta).
\end{aligned}
$$

As each parameter in Σ can move independently in the SGA algorithm, we instead use the lower triangluar decomposition $LL^T = \Sigma$. This results in the vector $\lambda = (\mu_1, \mu_2, L_{11}, L_{21}, L_{22}, \alpha, \beta)'$, where

	mu1	mu2	L11	L21	L22	alpha	beta	ELBO	Runtime (seconds)
Mean Field	0.83	0.1	0.04	0	0.05	50	79.7	-77.7	1.3 x 10^{-6}
SGA	0.63	0.27	0.09	-0.12	0.05	45.6	81.4	-78.3	51.1
SGA-RP	0.78	0.15	0.1	-0.09	0.05	49.3	78	-76.7	4
SGA-CV	0.77	0.15	0.09	-0.09	0.05	48.5	77.6	-76.9	1.3

Table 1: Parameter estimates, maximised ELBO and runtime of all Variational Bayes algorithms.

- μ_1: The mean of ϕ_1,
- μ_2: The mean of ϕ_2,
- L_{11}, L_{21}, L_{22}, the three components of L, the lower triangular decomposition of Σ,
- α, β, the shape and rate parameters of an Inverse Gamma distribution for σ^2.

The diagonal of L is not forced to be positive so the decomposition is not invariant to sign changes.

We compare three forms of the SGA algorithm, the original algorithm described by **??**, the reparameterised version described by **??** and the control variate version described by **??**. Each version is implemented with $\rho^(m)$ using AdaGrad as in (**??**.

The reparameterised SGA algorithm requires an auxillay distribution $q(\epsilon)$ and transformation. We use $(\epsilon_1, \epsilon_2)' \sim \mathcal{N}(0, I)$ and $\epsilon_3 \sim \mathcal{U}(0, 1)$. Then make the transformation

$$\begin{aligned}\phi_1 &= \mu_1 + L_1 1 \epsilon_1 \\ \phi_2 &= \mu_2 + L_2 1 \epsilon_1 + L_2 2 \epsilon_2 \\ \sigma^2 &= Q^{-1}(\epsilon_3, \alpha, \beta)\end{aligned}$$

where $Q^{-1}(\epsilon_3, \alpha, \beta)$ is the Inverse Gamma distribution inverse cdf function with shape α and scale β.

We summarise each algorithm below in table **??** by its estimated parameters, converged value of the ELBO and runtime. Mean Field algorithm runs extremely quickly but can not estimate the dependence between paramters so the converged ELBO is lower than the alternatives, except for the original SGA algorithm. This estimator is extremely noisy and required a value of S of 200, which slowed computation but it still converged to a lower optima than the variance reduced algorithms. The reparameterised algorithm, SGA-RP, could be run with $S = 5$ leading to its extremely fast runtime. The control variate algorithm, SGA-CV, was run with $S = 50$, and used 20 draws per iteration to estimate $\hat{a}$.

3 Approximate Forecast Updating

3.1 Motivation

-Slow convergence of MCMC -Bad scaling to large dimensional models and datasets -Time/Computational Constraints -Motivation from short forecast windows

3.2 Model Selection

-Overnight/Intermittent MCMC -Infer model structure from draws -Have easy parametric form of posterior to work with

4 Appendix

4.1 AR2 Model