"On Variational Bayesian Methods"
Draft Conceptual Notes

2 December 2025

1 Variational Bayes, KL Divergence, and the ELBO

My current understanding is that Variational Bayes (VB) is an optimisation-based approach to approximate Bayesian inference. The starting point is a posterior distribution

$$p(\theta \mid y)$$

that is conceptually well-defined but not easy to work with directly (for example, it may not have a closed form, or it may be expensive to sample from with MCMC).

VB introduces a simpler, tractable family of distributions

$$\{q_\lambda(\theta) : \lambda \in \Lambda\},$$

and then chooses the member of this family that is "closest" to the true posterior in Kullback–Leibler (KL) divergence. The KL divergence from q_λ to the posterior is

$$\mathsf{KL}\big(q_\lambda(\theta) \,\|\, p(\theta \mid y)\big) = \int q_\lambda(\theta)\, \log\left\{\frac{q_\lambda(\theta)}{p(\theta \mid y)}\right\} d\theta.$$

Minimising this KL divergence over λ selects the best approximating distribution q_λ within the chosen family.

In practice, we do not minimise the KL divergence directly. Instead, we work with the Evidence Lower BOund (ELBO), defined as

$$\mathcal{L}(\lambda) = E_{q_\lambda}[\log p(y, \theta)] - E_{q_\lambda}[\log q_\lambda(\theta)].$$

It can be shown that the log marginal likelihood decomposes as

$$\log p(y) = \mathcal{L}(\lambda) + \mathsf{KL}\big(q_\lambda(\theta) \,\|\, p(\theta \mid y)\big).$$

For fixed data y, $\log p(y)$ does not depend on λ. This means that maximising the ELBO $\mathcal{L}(\lambda)$ is exactly equivalent to minimising the KL divergence. Conceptually, KL divergence provides the "distance" between q_λ and the true posterior; operationally, the ELBO is the quantity we actually compute and maximise.

In the report, I plan to keep the algebra for this section fairly light and support it with one diagram showing the ELBO as a lower bound on $\log p(y)$, and the idea that pushing the bound up reduces the KL divergence between the approximation and the true posterior.

2 Mean-Field and Fixed-Form Variational Bayes

My current understanding is that VB can be organised into two main flavours, which differ in how the approximating family q_λ is chosen and how the optimisation is carried out.

Mean-Field Variational Bayes (MFVB)

In Mean-Field VB, we impose a factorisation assumption on the approximating distribution. Suppose the parameter can be partitioned as $\theta = (\theta_1, \dots, \theta_K)$. The Mean-Field assumption writes

$$q(\theta) = \prod_{k=1}^{K} q_k(\theta_k),$$

so the joint approximation is a product of simpler "blocks". When the model is in a conjugate-exponential family, this factorisation leads to closed-form coordinate-ascent updates for each factor q_k.

The standard Mean-Field recipe is:

- derive the full conditional for each block θ_k up to proportionality;
- take expectations with respect to the current q for all other blocks;
- update q_k using this expected log joint;
- cycle through the blocks until the ELBO stabilises.

In the linear regression example, the key idea is that we can treat the regression coefficients as one block (or a small number of blocks) rather than splitting them too finely. This blocking affects the approximate posterior covariance structure and therefore the degree of under- or over-dispersion.

Fixed-Form Variational Bayes (FFVB)

In Fixed-Form VB, we explicitly choose a parametric family $q_\lambda(\theta)$ in advance, such as a multivariate normal with either a full covariance matrix or some structured factor covariance. The goal is then to maximise the ELBO with respect to the parameters λ.

Here the updates are not usually available in closed form. Instead, modern FFVB uses gradient-based methods:

- stochastic gradient ascent on $\mathcal{L}(\lambda)$;
- natural gradients that respect the geometry of the variational family;
- reparameterisation tricks to obtain low-variance gradient estimates.

The main conceptual point I want to convey is that:

- Mean-Field VB looks "Gibbs-like", with coordinate updates derived from conjugacy;
- Fixed-Form VB makes it clear that VB is a general optimisation framework over an approximating family q_λ.

In the report I plan to keep the FFVB section short and non-technical, mainly to position VB as an optimisation method rather than to implement a full FFVB scheme.

3 Under-Dispersion and Calibration

A recurring theme in the VB literature, and in your comments, is that Mean-Field VB often underestimates posterior uncertainty. My current understanding is:

- Because the KL divergence used in standard VB penalises placing mass where the true posterior has little mass, the approximation tends to "hug" high-density regions and can neglect the tails.
- The Mean-Field factorisation removes dependence between blocks of parameters. This can artificially reduce posterior variances compared with a full, joint posterior that allows correlations.

In practice, this shows up as:

- posterior variances from VB that are smaller than those from a well-converged MCMC baseline;
- credible intervals that are too narrow, even when point estimates or predictive means look reasonable.

To make this concrete in the regression examples, I plan to:

- compute **variance ratios** for key parameters:

$$\text{Variance ratio} = \frac{\mathsf{Var}_{\mathsf{VB}}(\theta_j)}{\mathsf{Var}_{\mathsf{MCMC}}(\theta_j)},$$

  with values below 1 indicating under-dispersion;
- compare **predictive calibration** in the logistic regression example using:
  - reliability (calibration) curves, where predicted probabilities are grouped into bins and compared to observed frequencies;
  - Brier scores as a scalar measure of calibration and accuracy.

If appropriate, I would also like to explore simple calibration adjustments:

- varying prior scales to see whether slightly wider priors can partially offset under-dispersion;
- temperature scaling of predictive probabilities for logistic regression as a simple post-hoc correction, without changing the underlying VB posterior.

The aim is not to "fix" under-dispersion completely, but to show clearly where VB does well (for example, in predictive means) and where it is over-confident compared with an MCMC benchmark.

4 Metrics for the Empirical Comparison

For the small empirical study, the goal is to compare VB and MCMC in terms of both predictive performance and uncertainty behaviour, using metrics that are familiar from regression and classification.

Predictive performance

- **Log-likelihood / log-score** For each observation, I will evaluate the log predictive density (or probability) at the observed outcome and average these values over the dataset. Higher average log-scores indicate that the method places more probability mass on what actually happened.
- **RMSE (linear regression)** In the linear regression example, I plan to compute the Root Mean Squared Error between predicted means and observed responses. This gives a scale-dependent measure of how far, on average, the predictions are from the truth.
- **Accuracy and AUC (logistic regression)** For the logistic regression example:
  - Accuracy will be the proportion of correctly classified observations after choosing a threshold (for example, 0.5).
  - The Area Under the ROC Curve (AUC) will summarise how well the model ranks positive cases above negative cases across all thresholds.

Uncertainty and calibration

- **Variance ratios (VB vs MCMC)** For selected parameters, I plan to report the ratio of VB posterior variance to MCMC posterior variance. This is intended as a simple summary of how much narrower the VB posterior is, relative to a baseline that we treat as closer to the truth.
- **Reliability curves and Brier scores** In the logistic regression example, I plan to:
  - plot reliability curves to compare predicted probabilities with observed frequencies in bins;
  - compute Brier scores (mean squared error between predicted probabilities and binary outcomes) as a scalar summary of calibration and accuracy.

Computational cost

- **Wall-clock time** For both VB and the MCMC baseline, I plan to record the elapsed (wall-clock) time required to fit each model and, if relevant, to generate predictions. The idea is to give at least a rough sense of how much speed is gained by using VB relative to a standard NUTS implementation, recognising that this will depend on the specific implementation and hardware.

Taken together, these metrics should allow a focused comparison: VB vs MCMC on predictive performance, on how uncertainty is represented (under-dispersion and calibration), and on practical runtime, without attempting a large or exhaustive simulation study.