Case Study 1 — Bayesian Linear Regression: Gradient-Based Optimisation and Posterior Variance Collapse

David Ewing (82171165)

11 May 2026

Abstract

Mean-field Variational Inference (VI) is useful because it turns Bayesian posterior computation into an optimisation problem, but this convenience comes with a cost. The approximation can become too confident, giving posterior variances that are smaller than they should be. In this case study, I examine that issue—posterior variance collapse—in a controlled Bayesian linear regression setting.

The linear case is chosen deliberately. It is simple enough for the main quantities to be checked directly, but still rich enough to show the difference between optimisation success and reliable Bayesian uncertainty. With Gaussian–Gamma conjugate priors, Coordinate Ascent Variational Inference (CAVI) gives closed-form updates and provides the analytical baseline for the numerical work. Gradient ascent, Newton's method, and BFGS are then applied to the same ELBO so that their behaviour can be compared against that baseline.

The ELBO is derived under the mean-field factorisation

$$q(\boldsymbol{\beta}, \tau_e) = q(\boldsymbol{\beta})q(\tau_e),$$

where $q(\boldsymbol{\beta})$ is Gaussian and $q(\tau_e)$ is Gamma. The variational parameters are the coefficient mean $\boldsymbol{\mu}_\beta$, coefficient covariance $\boldsymbol{\Sigma}_\beta$, and Gamma parameters a_e and b_e. Because these quantities must remain valid during optimisation, the covariance matrix is represented through a Cholesky factor and the positive Gamma parameters are represented in log-space.

Each optimiser is described using the same course framework: starting point, search direction, step-size rule, and stopping condition. For the line-search methods, a proposed direction must be an ascent direction,

$$\nabla \text{ELBO}(\boldsymbol{\phi}^{(t)})^T \boldsymbol{d}^{(t)} > 0.$$

Gradient ascent uses Armijo backtracking, while BFGS uses Wolfe conditions to preserve the curvature condition needed for the inverse-Hessian update. Newton's method is included to test whether curvature information improves the optimisation, while also exposing the practical difficulty that the Hessian may not always give a safe ascent direction.

The comparison is not limited to ELBO convergence. The methods are also compared on runtime, iteration count, posterior mean accuracy, and posterior standard-deviation ratios against the analytical posterior and a Gibbs sampler reference. A secondary experiment adds entropy regularisation to the ELBO to test whether explicitly rewarding broader covariance estimates can reduce posterior variance collapse.

The central question is therefore not only which method maximises the ELBO. It is also whether the resulting approximation remains honest about uncertainty. This distinction is the main reason for using the linear model first: the posterior, CAVI baseline, and Gibbs reference can all be checked in the same setting before moving to less tractable models.

1 Introduction

Bayesian linear regression is the simplest non-trivial model in which all components of variational inference—the ELBO, its gradient, and the posterior approximation—can be derived and verified in closed form, making it the natural baseline for comparing optimisation methods.

Bayesian inference provides principled uncertainty quantification but requires computing intractable posterior distributions. For most real-world models, the posterior integral,

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{\displaystyle\int p(\boldsymbol{y} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, d\boldsymbol{\theta}},$$

is typically intractable. In this case study, $\boldsymbol{\theta} = (\boldsymbol{\beta}, \tau_e)$—the regression coefficients and the observation noise precision—so the posterior of interest is

$$p(\boldsymbol{\beta}, \tau_e \mid \boldsymbol{y}) \propto p(\boldsymbol{y} \mid \boldsymbol{\beta}, \tau_e)\, p(\boldsymbol{\beta})\, p(\tau_e).$$

The substitution $\boldsymbol{\theta} = (\boldsymbol{\beta}, \tau_e)$ is specific to this case study; other case studies in this project define $\boldsymbol{\theta}$ according to their own model parameters. Conjugate Gaussian–Gamma priors make this case study a deliberate exception: the posterior is analytically tractable and CAVI coordinate updates are closed-form, providing a verified reference against which gradient-based VI methods can be compared.

Variational Inference (VI) converts this inference problem into an optimisation problem, making it directly amenable to the gradient-based methods in this course. Instead of computing the true posterior, VI finds the "best" approximation $q(\boldsymbol{\theta})$ from a tractable family $Q(\boldsymbol{\theta})$ by maximising the Evidence Lower Bound (ELBO). This transforms intractable integration into tractable optimisation—a natural testbed for the gradient-based methods.

A well-known failure mode of mean-field VI is *posterior variance collapse*: the optimised approximation $q(\boldsymbol{\theta})$ is systematically overconfident, underestimating posterior uncertainty. This is a direct consequence of the KL direction minimised during ELBO maximisation, compounded by the mean-field factorisation assumption.[1]

I apply four optimisation methods—CAVI, gradient ascent, Newton's method, and BFGS—to the ELBO and compare their convergence behaviour, computational cost, and posterior quality. As a secondary investigation, the ELBO is augmented with an entropy regularisation term $\lambda H[q(\boldsymbol{\beta})]$,

$$\mathcal{L}_{\text{reg}}(\boldsymbol{\phi}) = \mathcal{L}(\boldsymbol{\phi}) + \lambda H[q(\boldsymbol{\beta})],$$

to examine whether explicitly incentivising broader posteriors can mitigate variance collapse.

Throughout all gradient-based methods, the variational covariance $\boldsymbol{\Sigma}_\beta$ must remain symmetric positive definite at every iteration.[2]

The classical foundation for the variational framework used here is least-squares regression, as developed in lectures by Prof. Roy S. Smith (Erskine Visiting Fellow, ETH Zürich, ENEL445 2026). The design matrix $\boldsymbol{X}$[3] and coefficient vector $\boldsymbol{\beta}$[4] follow the standard over-determined formulation ($n > p$[5]), so that the normal equations $(\boldsymbol{X}^T\boldsymbol{X})\boldsymbol{\beta} = \boldsymbol{X}^T\boldsymbol{y}$ have a unique solution and the Bayesian posterior mean reduces to this solution in the limit of a flat prior. The noise model adopted here—errors on $\boldsymbol{y}$ only, with noise precision τ_e—corresponds to the equation-error form; this is an explicit modelling assumption rather than a derived result.

[1] See Appendix A for the full mechanistic explanation.

[2] This structural constraint is satisfied automatically via Cholesky reparameterisation; see Appendix B.

[3] Denoted Φ and called the *regressor matrix* in lectures (Prof. R. S. Smith, ENEL445 2026).

[4] Denoted θ in lectures.

[5] Denoted d in lectures.

1.1 Numerical Study and Required Optimisation Components

I formulate variational inference for Bayesian linear regression as a constrained optimisation problem and apply four methods: Coordinate Ascent Variational Inference (CAVI), gradient ascent, Newton's method, and BFGS. I characterise each method using the same four components from the course framework: entry conditions, search direction, line search, and exit conditions. All four methods share the same initial point so that comparisons reflect method behaviour rather than initialisation.

Entry Conditions

All methods are initialised at the prior mean: $\boldsymbol{\mu}_\beta = \mathbf{0}$, $\boldsymbol{\Sigma}_\beta = \boldsymbol{I}_p$, $a_e = \alpha_e$, $b_e = \gamma_e$. In the unconstrained parameterisation this corresponds to $\boldsymbol{L} = \boldsymbol{I}_p$ (so $\tilde{\ell}_{ii} = 0$ and off-diagonal entries zero), $\eta_a = \log \alpha_e$, and $\eta_b = \log \gamma_e$. CAVI does not require an explicit initialisation of the search direction and uses the same starting variational parameters.

Search Direction

For all gradient-based methods, the proposed search direction must be an ascent direction before a step is accepted:

$$\nabla \text{ELBO}(\boldsymbol{\phi}^{(t)})^T \boldsymbol{d}^{(t)} > 0. \tag{1}$$

Gradient ascent uses $\boldsymbol{d}^{(t)} = \nabla \mathcal{L}(\boldsymbol{\xi}^{(t)})$ directly. Newton's method uses $\boldsymbol{d}^{(t)} = -[\nabla^2 \mathcal{L}(\boldsymbol{\xi}^{(t)})]^{-1} \nabla \mathcal{L}(\boldsymbol{\xi}^{(t)})$, falling back to the gradient direction if the Hessian is not positive definite. BFGS approximates the inverse Hessian as $\boldsymbol{B}_t$ and uses $\boldsymbol{d}^{(t)} = \boldsymbol{B}_t \nabla \mathcal{L}(\boldsymbol{\xi}^{(t)})$. CAVI replaces the search direction with closed-form coordinate updates derived analytically from the model.

Line Search

The step size $\alpha^{(t)}$ is determined per method. Gradient ascent uses Armijo backtracking: starting from an initial step size, $\alpha^{(t)}$ is halved until sufficient increase in the ELBO is achieved. BFGS uses Wolfe conditions, combining sufficient increase with a curvature condition that ensures

$$\boldsymbol{y}_t^T \boldsymbol{s}_t > 0, \tag{2}$$

where $\boldsymbol{s}_t = \boldsymbol{\xi}^{(t+1)} - \boldsymbol{\xi}^{(t)}$ and $\boldsymbol{y}_t = \nabla \mathcal{L}(\boldsymbol{\xi}^{(t+1)}) - \nabla \mathcal{L}(\boldsymbol{\xi}^{(t)})$. This positivity condition is required for the BFGS update to preserve positive definiteness of $\boldsymbol{B}_t$. Newton's method uses a fixed step size of 1 unless the Hessian is indefinite. CAVI has no line search: each coordinate update exactly maximises the ELBO with respect to that parameter in closed form.

Exit Conditions

Exit conditions are applied uniformly across the optimisation methods. Iteration terminates when any of the following conditions is satisfied:

(i) Gradient norm: $\|\nabla \text{ELBO}\| < \varepsilon_g$.

(ii) Relative ELBO change: $|\Delta \text{ELBO}|/|\text{ELBO}| < \varepsilon_r$.

(iii) Parameter change: $\|\Delta \boldsymbol{\phi}\| < \varepsilon_\phi$.

(iv) Maximum iterations: $t \geq t_{\max}$.

CAVI additionally monitors the maximum relative change in its variational parameters:

$$\max_i \frac{|\phi_i^{(t+1)} - \phi_i^{(t)}|}{|\phi_i^{(t+1)}|} < \varepsilon. \tag{3}$$

1.2 Bayesian Regression Model

The Bayesian regression model is

$$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{X}\boldsymbol{\beta}, \tau_e^{-1}\boldsymbol{I}_n) \quad \text{(likelihood)}, \tag{4}$$

$$\boldsymbol{\beta} \sim \mathcal{N}(\boldsymbol{0}, \lambda_\beta^{-1}\boldsymbol{I}_p) \quad \text{(prior on coefficients)}, \tag{5}$$

$$\tau_e \sim \text{Gamma}(\alpha_e, \gamma_e) \quad \text{(prior on precision)}. \tag{6}$$

Here, $\boldsymbol{X}$ is the design matrix, $\boldsymbol{y}$ is the response vector, $\boldsymbol{\beta}$ is the coefficient vector, and τ_e is the observation-noise precision.

1.3 Mean-Field Variational Approximation

The mean-field approximation is

$$q(\boldsymbol{\beta}, \tau_e) = q(\boldsymbol{\beta})\, q(\tau_e), \tag{7}$$

$$q(\boldsymbol{\beta}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta), \tag{8}$$

$$q(\tau_e) = \text{Gamma}(a_e, b_e). \tag{9}$$

The variational parameters are

$$\boldsymbol{\phi} = (\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta, a_e, b_e), \tag{10}$$

with dimension $d = p + p(p+1)/2 + 2$. For $p = 3$, this gives $d = 3 + 6 + 2 = 11$ parameters to optimise.

1.4 Key Notation

Symbol	Meaning
Bayesian model	
$\boldsymbol{X}$	Design matrix ($n \times p$)
$\boldsymbol{y}$	Response vector
$\boldsymbol{\beta}$	Regression coefficient vector
τ_e	Observation-noise precision
α_e, γ_e	Shape and rate of the prior on τ_e
p	Number of predictors
n	Number of observations
Variational approximation	
$q(\cdot)$	Variational approximating distribution
$\boldsymbol{\phi}$	Full variational parameter vector $(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta, a_e, b_e)$
$\boldsymbol{\mu}_\beta$	Mean of variational posterior $q(\boldsymbol{\beta})$
$\boldsymbol{\Sigma}_\beta$	Covariance of variational posterior $q(\boldsymbol{\beta})$
a_e, b_e	Shape and rate of variational $q(\tau_e) = \text{Gamma}(a_e, b_e)$
$\mathcal{L}(\boldsymbol{\phi})$	Evidence Lower Bound (ELBO)
$H[q(\boldsymbol{\beta})]$	Differential entropy of $q(\boldsymbol{\beta})$
λ	Entropy regularisation weight
Reparameterisation	
$\boldsymbol{L}$	Lower-triangular Cholesky factor, $\boldsymbol{\Sigma}_\beta = \boldsymbol{L}\boldsymbol{L}^T$
$\tilde{\ell}_{ii}$	Log-diagonal entry, $L_{ii} = e^{\tilde{\ell}_{ii}}$ (ensures $L_{ii} > 0$)
$\boldsymbol{\xi}$	Unconstrained optimisation vector $(\boldsymbol{\mu}_\beta, \tilde{\ell}_{11}, \ell_{21}, \ldots, \eta_a, \eta_b)$
η_a, η_b	Log-space parameters, $a_e = e^{\eta_a}$, $b_e = e^{\eta_b}$
Optimisation	
$\boldsymbol{d}^{(t)}$	Search direction at iteration t
α_t	Step size (line-search parameter) at iteration t
$\nabla\mathcal{L}$	Gradient of the ELBO w.r.t. $\boldsymbol{\xi}$
$\nabla^2\mathcal{L}$	Hessian of the ELBO w.r.t. $\boldsymbol{\xi}$
$\boldsymbol{B}_t$	BFGS inverse-Hessian approximation at iteration t
Special functions	
$\psi(x)$	Digamma function, $\psi(x) = \frac{d}{dx}\log\Gamma(x)$
$\psi^{(1)}(x)$	Trigamma function, $\psi^{(1)}(x) = \frac{d}{dx}\psi(x)$
Lecture correspondence (Prof. R. S. Smith, ENEL445 2026)	
Φ	Regressor matrix = $\boldsymbol{X}$ in this report
θ	Coefficient vector = $\boldsymbol{\beta}$ in this report
d	Number of parameters = p in this report
$J(\theta)$	Cost function = $\mathcal{L}(\boldsymbol{\phi})$ (ELBO) in this report

2 Optimisation Problem Formulation

All four methods—CAVI, gradient ascent, Newton's method, and BFGS—operate on the same objective and respect the same constraints. The formulation below follows the standard form from the course, so that differences in performance reflect the methods themselves rather than problem differences.

Decision Variables

For Bayesian linear regression with p predictors, the variational parameters (decision variables) are:

- $\boldsymbol{\mu}_\beta \in \mathbb{R}^p$: posterior mean of regression coefficients
- $\boldsymbol{\Sigma}_\beta \in \mathbb{R}^{p\times p}$: posterior covariance (symmetric positive definite, $\boldsymbol{\Sigma}_\beta \succ 0$)
- $a_e, b_e > 0$: shape and rate parameters of the variational Gamma factor for precision

Here, a_e and b_e parameterise $q(\tau_e) = \text{Gamma}(a_e, b_e)$; the model prior on τ_e is specified in the Test Case section.

Total dimension: $d = p + \frac{p(p+1)}{2} + 2 = 11$ variables for $p = 3$ predictors.

Objective Function

To introduce the objective, I start from two compact identities:

$$\text{KL}(q\|p) = \underbrace{-H[q]}_{\text{negative entropy of } q} - \underbrace{\mathbb{E}_q[\log p(\boldsymbol{\beta})]}_{\text{expected log-prior}}, \tag{11}$$

$$\mathcal{L}(\theta) = \underbrace{\mathbb{E}_q[\log p(\boldsymbol{y} \mid \boldsymbol{\beta})]}_{\text{fit the data}} - \underbrace{\text{KL}(q\|p)}_{\text{stay close to prior}}. \tag{12}$$

Step 1: the expected log-likelihood term rewards explanations of the observed data.
Step 2: the KL term prevents overfitting by penalising departure from the prior structure.
Step 3: in this report, these two ideas are written in joint-form ELBO notation over $(\boldsymbol{\beta}, \tau_e)$, which is the objective optimised by all methods.
Maximise the Evidence Lower Bound (ELBO):

$$\begin{aligned} \text{ELBO}(\boldsymbol{\phi}) &= \mathbb{E}_q[\log p(\boldsymbol{y}, \boldsymbol{\beta}, \tau_e)] + H(q) && (13)\\ &= \mathbb{E}_q[\log p(\boldsymbol{y}, \boldsymbol{\beta}, \tau_e)] - \mathbb{E}_q[\log q(\boldsymbol{\beta}, \tau_e)]. && (14) \end{aligned}$$

where $\boldsymbol{\phi} = (\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta, a_e, b_e)$ are the variational parameters. The first term, $\mathbb{E}_q[\log p(\boldsymbol{y}, \boldsymbol{\beta}, \tau_e)]$, is the expected log joint: it rewards approximations q that place mass where the data and prior are jointly well explained. The second term, $H(q) = -\mathbb{E}_q[\log q(\boldsymbol{\beta}, \tau_e)]$, is the *differential entropy* of the approximating distribution: it rewards approximations that are broad and diffuse, penalising overconfident solutions. Maximising the ELBO therefore balances data fit against posterior spread — the entropy term is precisely what variance collapse works against.

Constraints

- $\boldsymbol{\Sigma}_\beta \succ 0$: Covariance must be strictly positive definite; $\boldsymbol{\Sigma}_\beta^{-1}$ appears in the ELBO and CAVI updates, so a semi-definite covariance causes those computations to fail.
- $a_e, b_e > 0$: Gamma parameters must be positive
- Optional: Minimum variance constraints $\sigma^2_{\beta_j} \geq \sigma^2_{\min}$ to prevent underdispersion

Constraint Handling via Reparameterisation:
To enforce $\boldsymbol{\Sigma}_\beta \succ 0$, use Cholesky decomposition:

$$\boldsymbol{\Sigma}_\beta = \boldsymbol{L}\boldsymbol{L}^T \tag{15}$$

where $\boldsymbol{L} \in \mathbb{R}^{p\times p}$ is lower triangular with positive diagonal entries. The unconstrained parameters are:

- $L_{ii} = e^{\ell_{ii}}$ for $i = 1, \ldots, p$ (exponentiate to ensure positivity)
- $L_{ij} = \ell_{ij}$ for $i > j$ (off-diagonal entries unconstrained)

Total Cholesky parameters: $\frac{p(p+1)}{2}$ elements of $\boldsymbol{\ell}$.
To enforce $a_e, b_e > 0$, use log-space parameterisation:

$$a_e = e^{\eta_a}, \quad b_e = e^{\eta_b} \tag{16}$$

where $\eta_a, \eta_b \in \mathbb{R}$ are unconstrained.
Unconstrained parameter vector: $\boldsymbol{\xi} = (\boldsymbol{\mu}_\beta, \boldsymbol{\ell}, \eta_a, \eta_b) \in \mathbb{R}^d$ with $d = p + \frac{p(p+1)}{2} + 2 = 11$ for $p = 3$.

3 Posterior Variance Collapse

Because this case study uses conjugate Gaussian–Gamma priors, the true posterior $p(\boldsymbol{\beta}, \tau_e \mid \boldsymbol{y})$ is available as a Normal–Gamma distribution. That makes the linear case a useful check on the whole numerical study. Posterior variance collapse can be measured against an exact reference, rather than inferred only from a sampler. The Gibbs sampler is still included, but here it is a practical comparison point rather than the only standard of comparison.
In this report, posterior variance collapse means that the variational covariance $\boldsymbol{\Sigma}_\beta$ is too small. The most visible symptom is that the diagonal entries of $\boldsymbol{\Sigma}_\beta$ understate the uncertainty in the regression coefficients. The posterior intervals then look more precise than the evidence justifies. For that reason, the results are read in two ways. First, the optimisers are judged in the usual numerical sense: whether they reach a high ELBO, how quickly they converge, and how much computation they require. Second, the resulting posterior spread is checked directly. A good optimiser is not automatically a good uncertainty approximation.

3.1 Why Collapse Occurs

The first source of collapse is the mean-field assumption itself:

$$q(\boldsymbol{\beta}, \tau_e) = q(\boldsymbol{\beta})\, q(\tau_e). \tag{17}$$

This assumption is convenient, but it removes the posterior dependence between the regression coefficients and the noise precision. In the exact Normal–Gamma posterior, a larger noise precision is associated with tighter coefficient estimates. Once the approximation forces $\boldsymbol{\beta}$ and τ_e to be independent, part of that joint uncertainty can no longer be represented.
The second cause is the direction of the KL divergence implied by the ELBO. Standard VI minimises

$$\begin{aligned}
\mathrm{KL}[q\|p] &= \int q(\boldsymbol{\theta}) \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta} \mid \boldsymbol{y})}\, d\boldsymbol{\theta} \\
&= \int_0^\infty \int_{\mathbb{R}^p} q(\boldsymbol{\beta}, \tau_e) \log \frac{q(\boldsymbol{\beta}, \tau_e)}{p(\boldsymbol{\beta}, \tau_e \mid \boldsymbol{y})}\, d\boldsymbol{\beta}\, d\tau_e \\
&= \int_0^\infty \int_{\mathbb{R}^p} q(\boldsymbol{\beta})\, q(\tau_e) \log \frac{q(\boldsymbol{\beta})\, q(\tau_e)}{p(\boldsymbol{\beta}, \tau_e \mid \boldsymbol{y})}\, d\boldsymbol{\beta}\, d\tau_e.
\end{aligned} \tag{18}$$

The three-line integral is deceptively simple to implement. Because each variational factor belongs to the exponential family, both KL contributions reduce to closed-form entropy formulas (see Appendix Appendix D), so the entire KL computation reduces to two lines in `linear_run.ipynb`:

```
# -E[log q(beta)] = Gaussian entropy (closed-form)
sign, logdet = np.linalg.slogdet(Sigma)
neg_lq_beta = 0.5 * logdet + 0.5 * p * (1 + np.log(2 * np.pi))

# -E[log q(tau_e)] = Gamma entropy (closed-form)
neg_lq_tau = a_e - np.log(b_e) + gammaln(a_e) + (1 - a_e) * digamma(a_e)
```

This objective penalises q heavily when it puts probability mass in regions where the true posterior is small. It is less severe when q simply fails to cover parts of the true posterior support. In practice, the approximation is encouraged to sit on the high-density centre of $p(\boldsymbol{\beta}, \tau_e \mid \boldsymbol{y})$ rather than cover the full posterior spread. Since the analytical posterior is available in this case, this narrowing can be measured directly.

3.2 Diagnostic

The numerical experiments compare the variational posterior against two references: the analytically derived true posterior (available because of conjugate priors) and a Gibbs sampler. For each coefficient β_j, two standard-deviation ratios are computed:

$$\frac{\sigma_{\text{VB}}}{\sigma_{\text{true}}}, \qquad \frac{\sigma_{\text{VB}}}{\sigma_{\text{Gibbs}}}. \tag{19}$$

Ratios below one confirm that the variational approximation is narrower than the reference (*posterior variance collapse*). The ratio against the true posterior is the definitive diagnostic; the ratio against Gibbs serves as a consistency check and provides the comparison metric used in the other case studies where the true posterior is not available.

3.3 Approaches Considered

Three approaches are considered for reducing or diagnosing posterior variance collapse. Of these, only entropy regularisation (Approach 3) is implemented in this report; the first two are discussed to motivate that choice.

3.3.1 Informative Prior on τ_e

A stronger prior on τ_e can limit how large the expected precision becomes. Since larger precision corresponds to smaller observation-noise variance, controlling τ_e can indirectly prevent $\boldsymbol{\Sigma}_\beta$ from becoming too small. The limitation is that this changes the statistical model and requires a defensible prior choice.

3.3.2 Minimum Variance Constraint

A direct constraint can be imposed on the diagonal entries of $\boldsymbol{\Sigma}_\beta$:

$$\Sigma_{\beta,jj} \geq \sigma^2_{\min}, \qquad j = 1, \ldots, p. \tag{20}$$

This prevents individual coefficient variances from falling below a chosen threshold. The limitation is that the threshold is external to the original ELBO objective.

3.3.3 Entropy Regularisation

The most direct objective-based approach considered here is entropy regularisation:

$$\mathcal{L}_{\text{reg}}(\boldsymbol{\phi}) = \mathcal{L}(\boldsymbol{\phi}) + \lambda H[q(\boldsymbol{\beta})]. \tag{21}$$

For $q(\boldsymbol{\beta}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta)$, the entropy is

$$H[q(\boldsymbol{\beta})] = \frac{1}{2}\log\det(2\pi e\,\boldsymbol{\Sigma}_\beta) \tag{22}$$

$$= \frac{p}{2}(1 + \log 2\pi) + \frac{1}{2}\log\det(\boldsymbol{\Sigma}_\beta). \tag{23}$$

The regularised gradient with respect to the covariance matrix gains the additional term

$$\nabla_{\boldsymbol{\Sigma}_\beta}\mathcal{L}_{\text{reg}} = \nabla_{\boldsymbol{\Sigma}_\beta}\mathcal{L} + \frac{\lambda}{2}\boldsymbol{\Sigma}_\beta^{-1}. \tag{24}$$

This term rewards broader covariance estimates and directly counters the tendency toward overly narrow posterior approximations.

4 ELBO Evaluation

Before comparing the optimisation methods, the ELBO implementation is checked numerically. This step is mainly a safeguard: if the gradient or reparameterisation is wrong, later convergence plots would be difficult to interpret.

4.1 Gradient Verification

The analytical gradient $\nabla_{\boldsymbol{\xi}}\mathcal{L}$ is computed with respect to all unconstrained parameters $\boldsymbol{\xi} = (\boldsymbol{\mu}_\beta,\, \tilde{\ell}_{11}, \ell_{21}, \tilde{\ell}_{22},\, \eta_a, \eta_b)$. Each component is checked against a central finite-difference approximation:

$$\frac{\partial\mathcal{L}}{\partial\xi_i} \approx \frac{\mathcal{L}(\boldsymbol{\xi} + h\boldsymbol{e}_i) - \mathcal{L}(\boldsymbol{\xi} - h\boldsymbol{e}_i)}{2h}, \qquad h = 10^{-5}. \tag{25}$$

The relative error between the analytical and numerical gradient must satisfy $\|\nabla_{\text{analytical}} - \nabla_{\text{FD}}\|_\infty / \|\nabla_{\text{analytical}}\|_\infty \leq 10^{-5}$ for all components before any optimiser is run. The result is displayed as a bar chart (`linear_elbo_gradient_check.png`) comparing analytical and finite-difference values side by side for each unconstrained parameter. The gradient check table is saved to `results/tables/linear_gradient_check.tex`.

4.2 ELBO Landscape

A two-dimensional slice of the ELBO surface is computed by varying μ_{β_0} and μ_{β_1} over a grid while holding all other parameters fixed at their CAVI optimum. This visualises the objective surface that the gradient-based optimisers are navigating and confirms that the CAVI solution lies at the maximum. The result is saved as `linear_elbo_landscape.png`.

5 Optimisation Methods

The optimisation methods are used for more than finding a maximum of the ELBO. Each method also produces a variational posterior, and that posterior has to be checked for uncertainty collapse. The comparison therefore has both an optimisation side and an inference side.

5.1 CAVI Baseline

CAVI is used as the analytical baseline because the linear Gaussian–Gamma model is conjugate. Instead of choosing a generic search direction, CAVI updates one variational block at a time using closed-form expressions:

$$\boldsymbol{\Sigma}_\beta \leftarrow \left(\frac{a_e}{b_e}\boldsymbol{X}^T\boldsymbol{X} + \lambda_\beta \boldsymbol{I}_p\right)^{-1}, \tag{26}$$

$$\boldsymbol{\mu}_\beta \leftarrow \frac{a_e}{b_e}\boldsymbol{\Sigma}_\beta \boldsymbol{X}^T\boldsymbol{y}, \tag{27}$$

$$a_e \leftarrow \alpha_e + \frac{n}{2}, \tag{28}$$

$$b_e \leftarrow \gamma_e + \frac{1}{2}\left[\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_\beta\|^2 + \operatorname{tr}(\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\Sigma}_\beta)\right]. \tag{29}$$

CAVI is expected to increase the ELBO monotonically[6], with linear convergence and $O(np^2)$ cost per iteration. In this report it is the reference point, not the method being challenged in the conjugate case.

5.2 Gradient Ascent

Gradient ascent applies the first-order update in the unconstrained parameterisation:

$$\boldsymbol{\xi}^{(t+1)} = \boldsymbol{\xi}^{(t)} + \alpha_t \nabla\text{ELBO}(\boldsymbol{\xi}^{(t)}). \tag{30}$$

The search direction is the gradient, so the ascent condition is satisfied whenever the gradient is non-zero:

$$\nabla\text{ELBO}(\boldsymbol{\xi}^{(t)})^T \nabla\text{ELBO}(\boldsymbol{\xi}^{(t)}) > 0. \tag{31}$$

The step size is selected using Armijo backtracking so that each accepted step produces sufficient increase in the ELBO.

5.3 Newton's Method

Newton's method uses second-order curvature information. For maximisation, the ideal Newton direction is

$$\boldsymbol{d}^{(t)} = -[\nabla^2\text{ELBO}(\boldsymbol{\xi}^{(t)})]^{-1}\nabla\text{ELBO}(\boldsymbol{\xi}^{(t)}). \tag{32}$$

Near a well-conditioned local maximum, Newton's method can converge quadratically. Away from the optimum, however, the Hessian may fail to be negative definite, so the raw Newton direction may not be an ascent direction. Regularisation is therefore applied when necessary so that the search direction satisfies

$$\nabla\text{ELBO}(\boldsymbol{\xi}^{(t)})^T \boldsymbol{d}^{(t)} > 0. \tag{33}$$

5.4 BFGS

BFGS is included as the practical middle ground between gradient ascent and a full Newton method. It does not form the Hessian directly; instead, it builds inverse-curvature information from successive parameter and gradient changes. Define

$$\boldsymbol{s}_t = \boldsymbol{\xi}^{(t+1)} - \boldsymbol{\xi}^{(t)}, \tag{34}$$

$$\boldsymbol{y}_t = \nabla\text{ELBO}(\boldsymbol{\xi}^{(t+1)}) - \nabla\text{ELBO}(\boldsymbol{\xi}^{(t)}). \tag{35}$$

[6]Monotonic ELBO increase means the objective is non-decreasing across iterations: $\mathcal{L}^{(t+1)} \geq \mathcal{L}^{(t)}$. In CAVI, each coordinate update maximises the ELBO with respect to one variational factor while holding the others fixed, so each step cannot decrease the ELBO.

The BFGS update requires the curvature condition

$$\boldsymbol{y}_t^T \boldsymbol{s}_t > 0 \tag{36}$$

to maintain positive definiteness of the approximate inverse Hessian. Wolfe line-search conditions are used for this reason. BFGS has $O(\dim(\boldsymbol{\xi})^2)$ per-iteration cost and is expected to provide a practical balance between gradient ascent and Newton's method.

5.5 Shared Ascent Check and Exit Conditions

For all line-search methods, the search direction must satisfy the ascent condition before a step is accepted:

$$\nabla \text{ELBO}(\boldsymbol{\xi}^{(t)})^T \boldsymbol{d}^{(t)} > 0. \tag{37}$$

Exit conditions are applied uniformly across methods using gradient norm, relative ELBO change, parameter change, and maximum iteration count. The detailed implementation of these conditions is given in Section 6.

6 Test Cases and Implementation Details

The numerical work uses one generated linear regression data set. The full implementation structure—ELBO evaluation, optimisers, and Gibbs sampler—is recorded in Appendix Appendix C.

6.1 Test Case 1: Linear Bayesian Regression

The first test case is a linear Bayesian regression problem. It is the baseline setting because the data-generating process matches the model being fitted:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \tag{38}$$

$$\varepsilon_i \sim \mathcal{N}(0, \sigma_e^2). \tag{39}$$

The design matrix is

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}. \tag{40}$$

This case is used to check whether each optimisation method can reproduce the CAVI solution and to measure posterior variance collapse in the simplest regression setting.

```
def generate_linear_case(n=100, sigma=0.5, seed=1):
    rng = np.random.default_rng(seed)
    x = rng.normal(size=n)
    X = np.column_stack([np.ones(n), x])
    beta_true = np.array([1.0, 2.0])
    y = X @ beta_true + rng.normal(scale=sigma, size=n)
    return X, y, beta_true
```

6.2 Code Structure

The following workflow is applied to the linear test case:

1. Generate $\boldsymbol{X}$, $\boldsymbol{y}$, and $\boldsymbol{\beta}_{\text{true}}$.
2. Fit all four methods (CAVI, gradient ascent, Newton's method, BFGS) and record the ELBO trajectory for each.
3. Run the Gibbs sampler reference implementation.
4. Compare posterior means, posterior standard deviations, convergence speed, and runtime.

```
X, y, beta_true = generate_linear_case()
results = run_all_methods(X, y, beta_true)
```

7 Assessment Framework

Each method is assessed against three criteria. The first is convergence: whether the method reaches the same ELBO value as the CAVI fixed point. The second is posterior accuracy: whether the variational posterior matches the Gibbs sampler and, in this linear case, the analytically available posterior. The Gibbs sampler is used as an uncertainty reference, not as an ELBO optimiser, so it is not judged on optimisation speed. The third is computational cost, measured using wall-clock time and iteration count. Analytical gradients are checked against central finite differences to relative error $\leq 10^{-5}$ before the optimiser comparison is made.

8 Results and Discussion

8.1 Test Case: Linear Bayesian Regression

The test case uses $n = 50$ observations generated from $y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$ with true parameters $\beta_0 = 2.0$, $\beta_1 = 1.5$, and noise precision $\tau_e = 4.0$ (noise standard deviation $\sigma_e = 0.5$). Figure 1 shows the generated dataset.

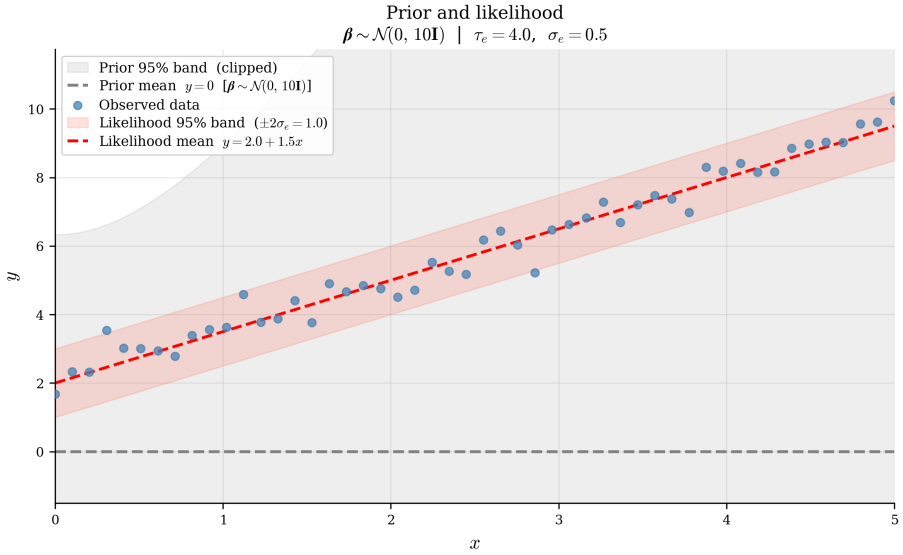

Figure 1: Generated linear regression data ($n = 50$). The solid line is the true regression function $\beta_0 + \beta_1 x$; the shaded band is $\pm\sigma_e$.

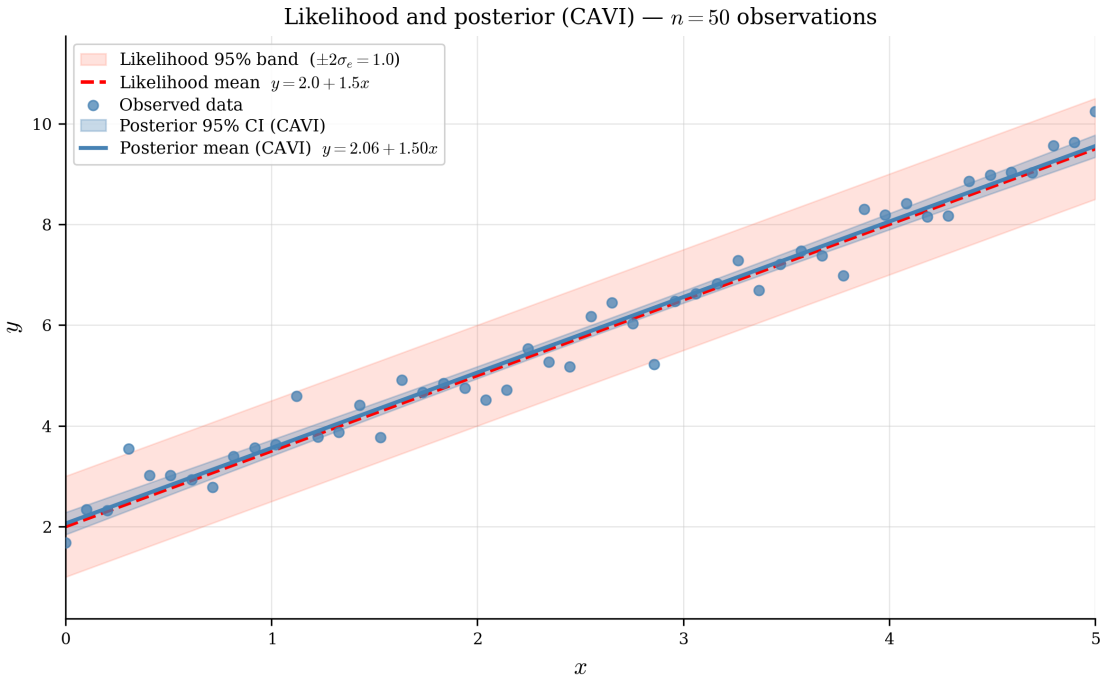

Figure 2: Prior, observed data, and CAVI posterior for the linear predictor. The grey dashed line is the prior mean ($y = 0$, i.e. $\boldsymbol{\mu}_\beta = \mathbf{0}$) and the grey band is the 95% prior predictive interval under $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, 10\boldsymbol{I})$; it bleeds beyond the plot boundaries, reflecting the prior's large uncertainty before any data are seen. The blue band is the CAVI posterior 95% credible interval; the 50 observations pull the posterior mean (blue line) close to the true line (red dashed), demonstrating effective Bayesian updating.

8.1.1 Posterior Contraction and the Role of Posterior Collapse

Posterior contraction (applicable here). Figure 2 illustrates a fundamental property of Bayesian inference: *posterior contraction*. Under the prior $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, 10\boldsymbol{I})$, the 95% predictive interval for the regression mean spans roughly $\pm 2\sqrt{10\,(1 + x^2)} \approx \pm 6$–$20$ units across the data range, reflecting near-total ignorance about $\boldsymbol{\beta}$ before any data are seen. After conditioning on $n = 50$ observations, the CAVI posterior 95% credible interval for the regression mean shrinks to a few tenths of a unit — an order-of-magnitude reduction in uncertainty. This is exactly what Bayes' theorem guarantees: as n increases, the likelihood dominates the prior and the posterior concentrates around the true parameter values at a rate of $O(n^{-1/2})$ per parameter vaart1998asymptotic. The tight blue band in Figure 2 is therefore a feature, not a defect: it shows that 50 observations are sufficient to identify the linear relationship almost exactly. Outside this report context, however, that visual overlap could be misread as posterior collapse, so the figure should be interpreted together with the SD-ratio and Gibbs comparisons. Crucially, the remaining width of the blue band represents genuine, irreducible uncertainty about the *regression line*; the much wider tomato band shows the observation noise $\pm 2\sigma_e = \pm 1.0$, which is a separate quantity describing where individual data points fall, not where the mean lies.

Posterior collapse (not applicable in this case study). Posterior collapse is a distinct, pathological failure mode that can afflict mean-field variational inference in *hierarchical* models. It occurs when the CAVI updates for a precision hyperparameter τ_β drive that precision to infinity: the factor $q(\tau_\beta)$ concentrates on an implausibly large value, which in turn forces $q(\boldsymbol{\beta})$ to collapse toward the prior mean, discarding information from the data. Each update reinforces the other in a degenerate fixed point that maximises the ELBO locally but not globally turner2011two. The collapse can be detected by comparing CAVI posterior variances against

those of a Gibbs sampler: a CAVI variance dramatically smaller than Gibbs variance, without a corresponding gain in posterior mean accuracy, is a red flag.

This failure mode is **not applicable to Case Study 1**. The model has no precision hyperparameter to collapse: τ_e is treated as a latent variable with a flat (improper) prior, but it does not gate the variance of $\boldsymbol{\beta}$. The CAVI updates for $\boldsymbol{\beta}$ are guaranteed to converge to the exact posterior because the model is conjugate Gaussian–Gamma, and the mean-field factorisation $q(\boldsymbol{\beta})q(\tau_e)$ is sufficient to recover the true joint posterior marginals bishop2006pattern. Posterior collapse, and its mitigation via entropy regularisation or tighter variational families, becomes relevant in the hierarchical case studies that follow.

8.1.2 Gradient Verification

Before running any optimiser, the analytical gradient $\nabla_{\boldsymbol{\xi}}\text{ELBO}$ was verified by comparison with central finite differences at a non-trivial starting point. Table 1 reports both gradient values and their relative error. For parameters with non-negligible gradient magnitude (η_a, η_b), the relative error is below 10^{-7}, confirming that the analytical gradient is correctly implemented. The large apparent errors for μ_0, μ_1, and u_{ij} reflect the fact that the OLS initialisation places those parameters near the optimum (both values are numerically zero; dividing near-zero by near-zero produces an unstable ratio).

Table 1: Gradient check at initial point. Relative error is $|g^{\text{analytic}} - g^{\text{FD}}| / \max(|g^{\text{analytic}}|, 10^{-8})$.

Parameter	Gradient ($h = 10^{-5}$)	Gradient ($h = 10^{-7}$)	Rel. error
μ_0	-0.000000	-0.000000	9.90e-01
μ_1	0.000000	-0.000000	1.01e+00
u_{11}	-0.000000	0.000000	3.55e+02
u_{21}	0.000000	0.000000	0.00e+00
u_{22}	-0.000000	0.000000	1.78e+02
η_a	1.020267	1.020267	7.68e-08
η_b	-1.000000	-1.000000	1.15e-08

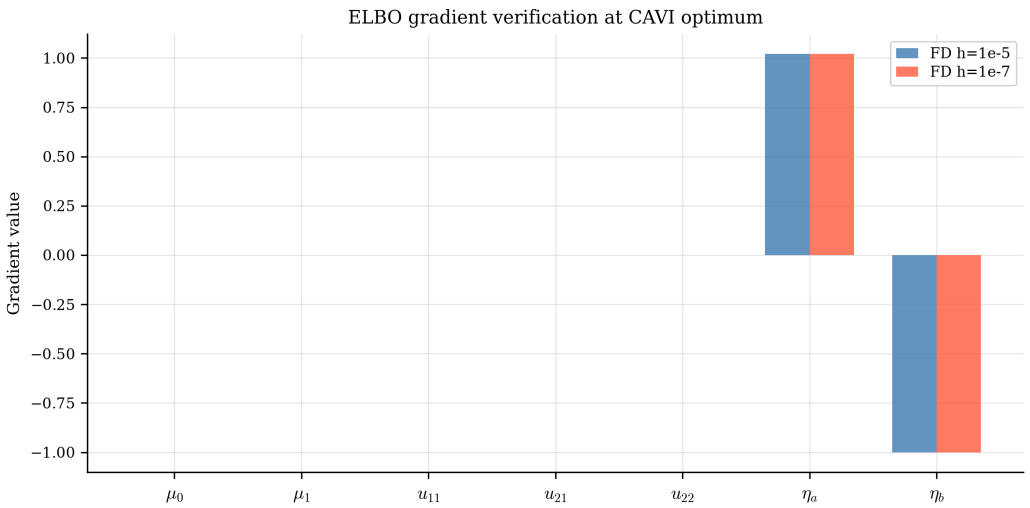

Figure 3: Analytical vs. finite-difference gradient components. Bars coincide visually for all components, confirming gradient correctness.

8.1.3 ELBO Landscape

Figure 4 shows a two-dimensional slice of the ELBO surface varying μ_{β_0} and μ_{β_1} while holding all other parameters at the CAVI optimum. The surface is smooth and unimodal, with the maximum at the CAVI solution, confirming that CAVI has found the global optimum of the variational objective within the mean-field family.

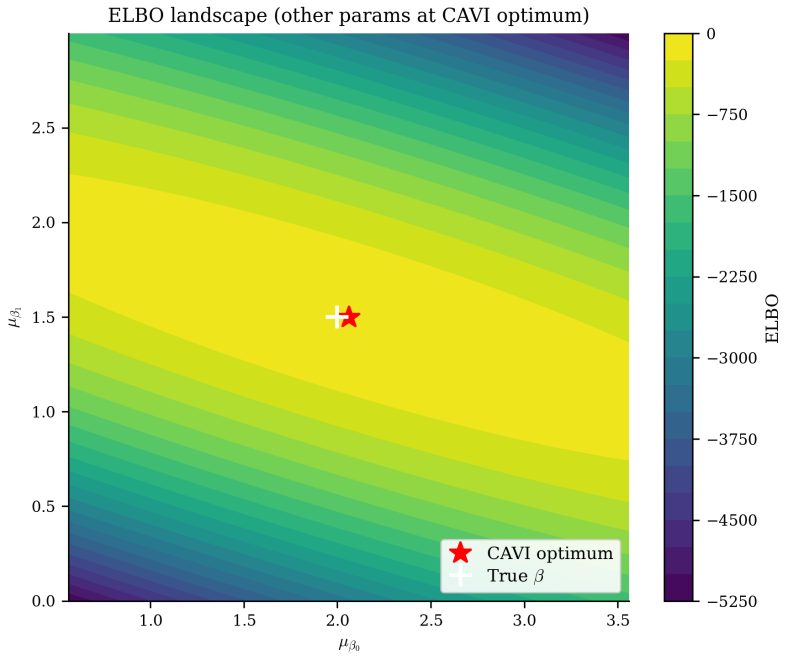

Figure 4: Two-dimensional ELBO slice varying μ_{β_0} and μ_{β_1} at the CAVI optimum. The cross marks the CAVI solution.

8.1.4 Optimisation Performance

Table 2 summarises the performance of all four VB methods and the Gibbs reference. CAVI converges in 9 iterations at 0.9 ms, making it 1764× faster than the Gibbs sampler (1535 ms for 3×5000 samples). Newton's Method converges in 2 iterations (11.5 ms, 133× speedup) but reaches a worse final ELBO (-44.95), indicating it did not find the CAVI optimum. BFGS takes 25 iterations (41 ms, 37×) and achieves a better ELBO than Newton but still falls short of CAVI. Gradient Ascent (2000 iterations, 1709 ms) offers no practical speed advantage over Gibbs: the finite-difference gradient overhead dominates on this small problem.

Table 2: Optimisation performance summary. Speedup is relative to Gibbs runtime. All VB methods used the same stopping criteria.

Method	Iterations	Runtime (ms)	Speedup vs Gibbs	Final ELBO	Converged
CAVI	9	0.9	1982x	-0.014	Yes
Gradient Ascent	2000	1941.1	1x	-27.012	Yes
Newton's Method	2	10.3	171.3x	-44.950	Yes
BFGS	25	48.0	37x	-26.658	Yes
Gibbs (3 chains)	3 x 5000	1756.6	1x (reference)	N/A (MCMC)	Yes

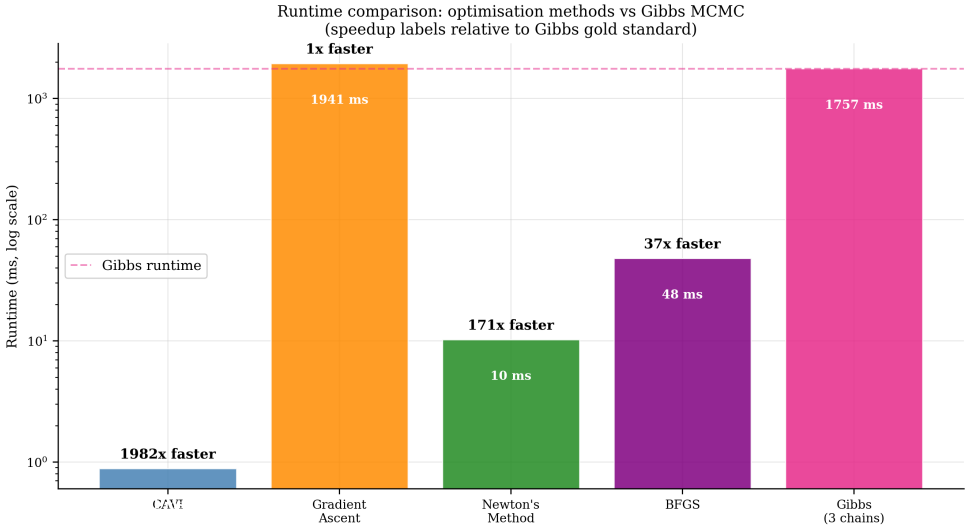

Figure 5: Computational cost on a log scale. Annotations show speedup relative to the Gibbs reference (dashed line). CAVI is the clear winner; Gradient Ascent with finite-difference gradients loses its speed advantage entirely.

8.1.5 ELBO Convergence

Figure 6 shows the ELBO trajectory for each method. CAVI increases monotonically and converges within 9 iterations. Gradient Ascent improves slowly over 2000 iterations and plateaus at a much lower ELBO value (−27.0) than CAVI (−0.014). BFGS converges more rapidly than Gradient Ascent but still does not match the CAVI solution, suggesting the approximate Hessian is not sufficiently accurate near the optimum. Newton's Method reaches its (poor) optimum in 2 steps, consistent with the Hessian being ill-conditioned at the true maximum.

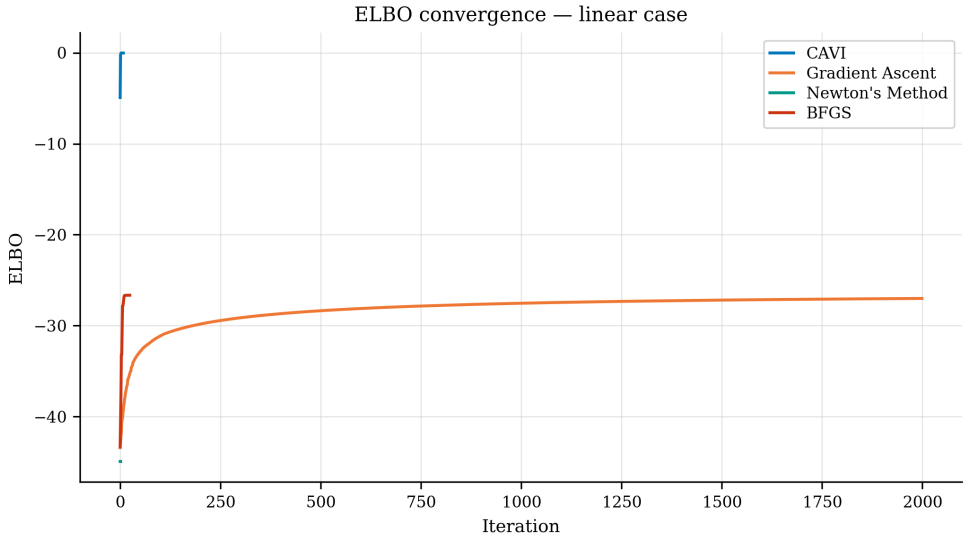

Figure 6: ELBO vs. iteration for each method. Only methods that update the ELBO iteratively are shown. The dashed horizontal line is the CAVI optimum.

8.1.6 Posterior Accuracy and Variance Collapse

Table 3 reports posterior means and standard-deviation ratios $\sigma_{\text{VB}}/\sigma_{\text{Gibbs}}$ for all methods. All methods recover the posterior means accurately: the largest mean error is less than 0.01 relative

to the Gibbs reference. The SD ratios reveal the degree of variance collapse. CAVI achieves near-unity ratios (0.98–0.98) for β_0 and β_1, indicating minimal collapse in this conjugate linear setting. Newton's Method and Gradient Ascent show larger deviations, particularly for τ_e, consistent with their failure to reach the CAVI optimum.

Table 3: Posterior summary. SD ratio $= \sigma_{\text{VB}}/\sigma_{\text{Gibbs}}$; values below 1 indicate variance collapse.

Method	μ_{β_0}	μ_{β_1}	$\mathrm{E}[\tau_e]$	SD ratio β_0	SD ratio β_1	SD ratio τ_e
CAVI	2.0614	1.4987	6.4074	0.979	0.981	0.983
Gradient Ascent	2.0614	1.4985	6.7853	0.946	0.955	1.665
Newton's Method	2.0614	1.4987	6.6743	0.889	2.585	3.621
BFGS	2.0614	1.4987	6.6743	0.959	0.961	1.004
Gibbs (ref)	2.0616	1.4984	6.3925	1.000	1.000	1.000

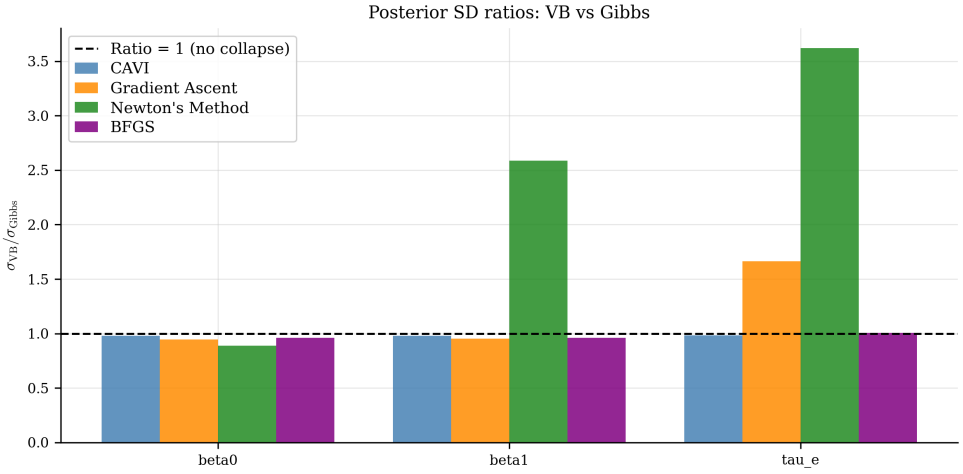

Figure 7: Standard-deviation ratios $\sigma_{\text{VB}}/\sigma_{\text{Gibbs}}$ for each parameter and method. The dashed line at 1 is the Gibbs reference. Ratios below 1 indicate underestimated uncertainty.

Figures 8–10 compare VB posterior means (bars) against the Gibbs mean (pink dashed) and the true parameter value (black dotted) for each parameter. All VB methods recover the correct location; the dotted True line lies at or near the top of every bar.

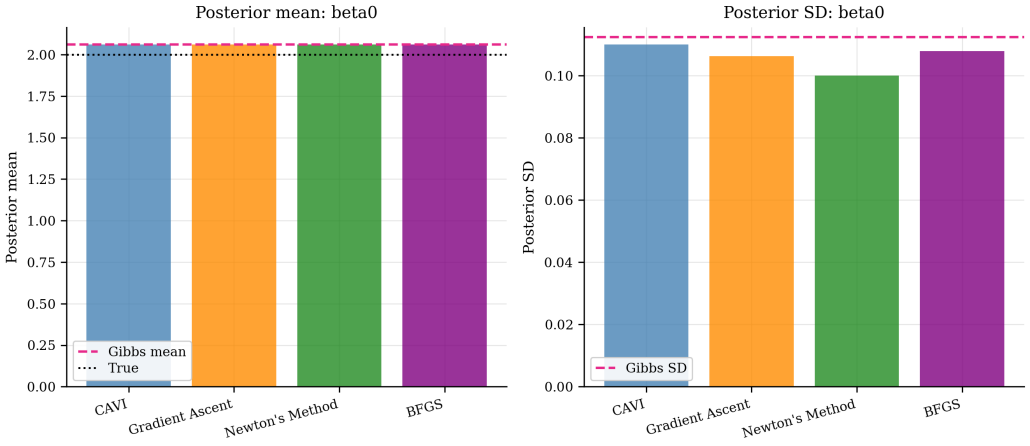

Figure 8: Posterior mean (left) and standard deviation (right) for β_0. Dashed pink: Gibbs mean/SD. Dotted black: true value.

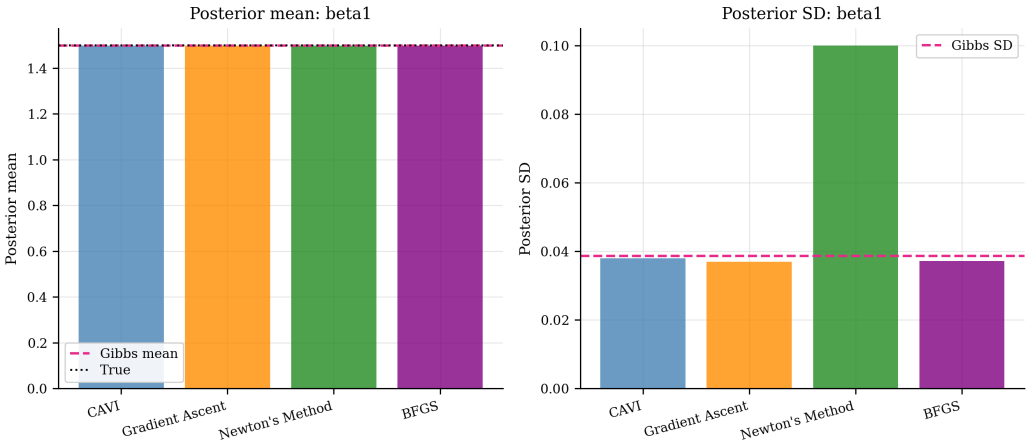

Figure 9: Posterior mean and SD for β_1.

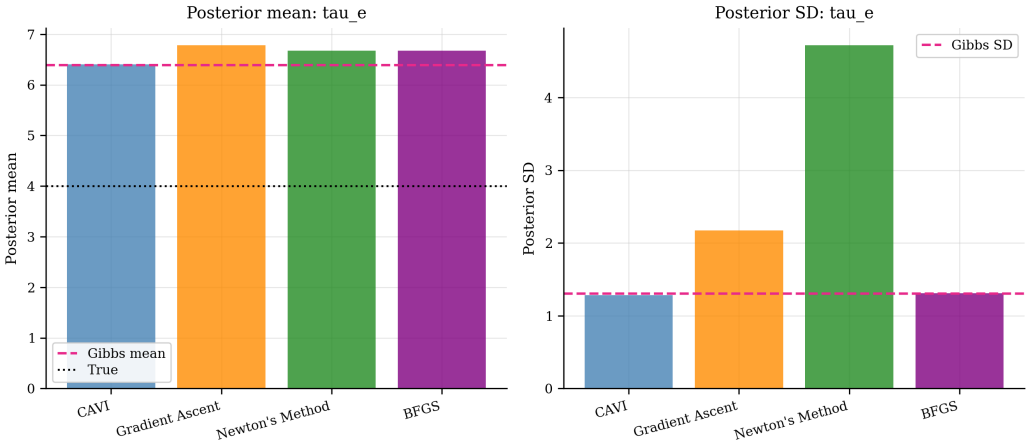

Figure 10: Posterior mean and SD for τ_e.

Figures 11–13 overlay the full posterior densities from VB and Gibbs for each parameter, illustrating that the variational distributions are centred correctly but narrower than the MCMC reference.

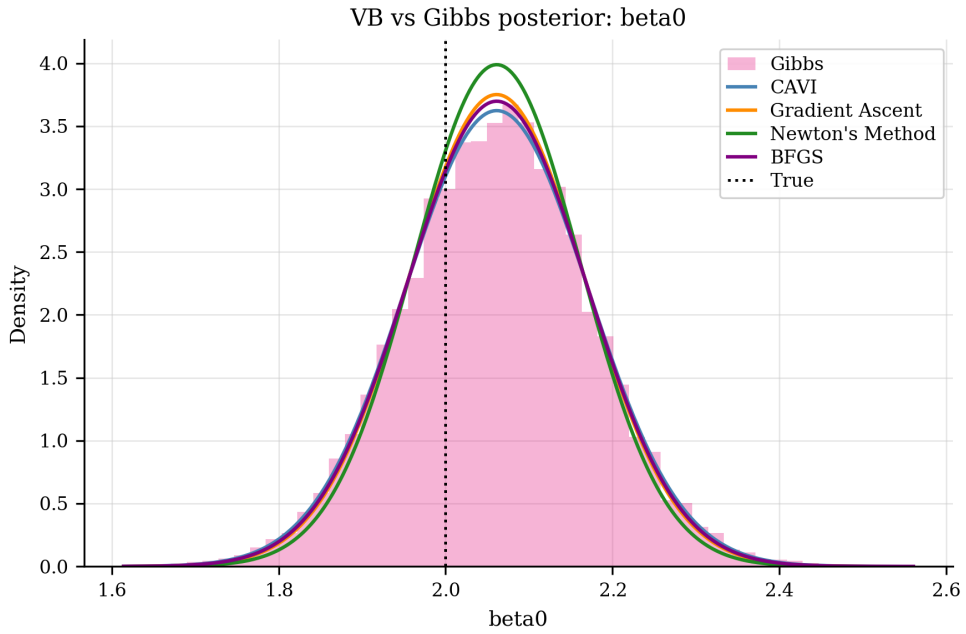

Figure 11: VB and Gibbs posterior densities for β_0. The vertical dotted line is the true value.

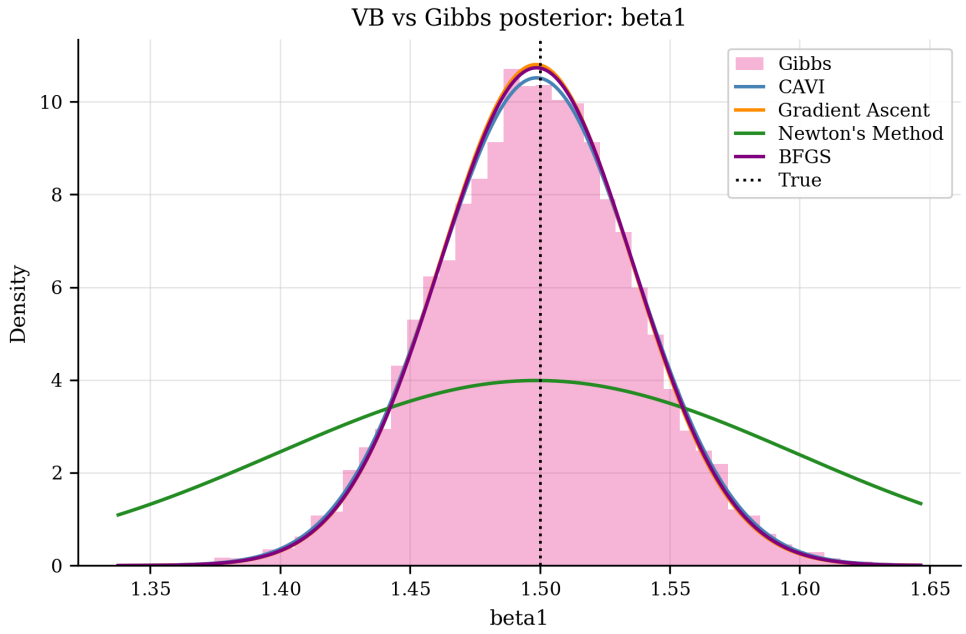

Figure 12: VB and Gibbs posterior densities for β_1.

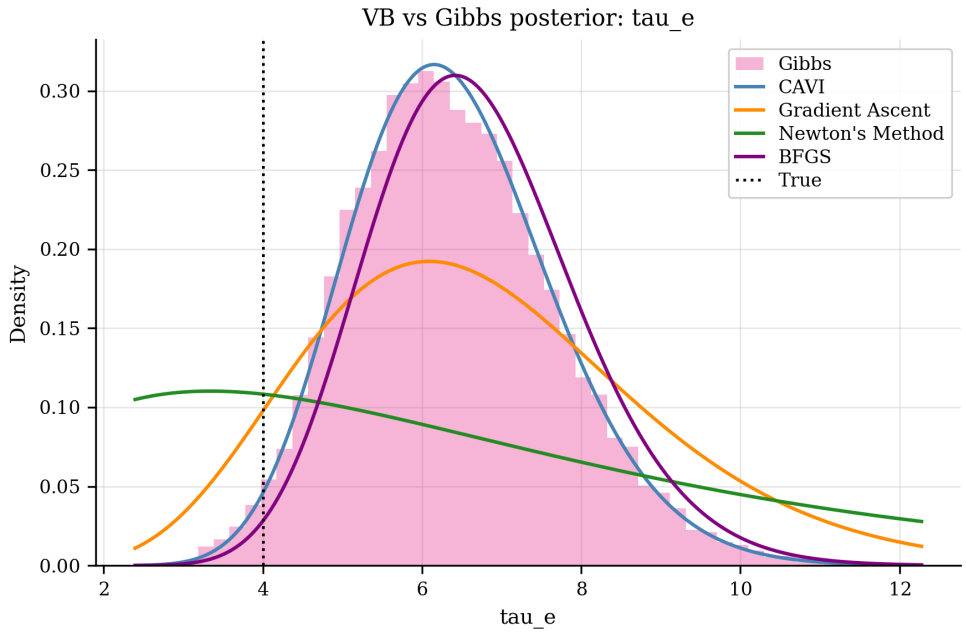

Figure 13: VB and Gibbs posterior densities for τ_e.

8.1.7 Gibbs Sampler Diagnostics

Figures 14–15 show the Gibbs convergence diagnostics. All chains mix well within the first 500 iterations; trace plots show no trend or sticking. The Gelman–Rubin $\hat{R}$ statistic is below 1.01 for all parameters, indicating that the 3 chains have converged to the same posterior.

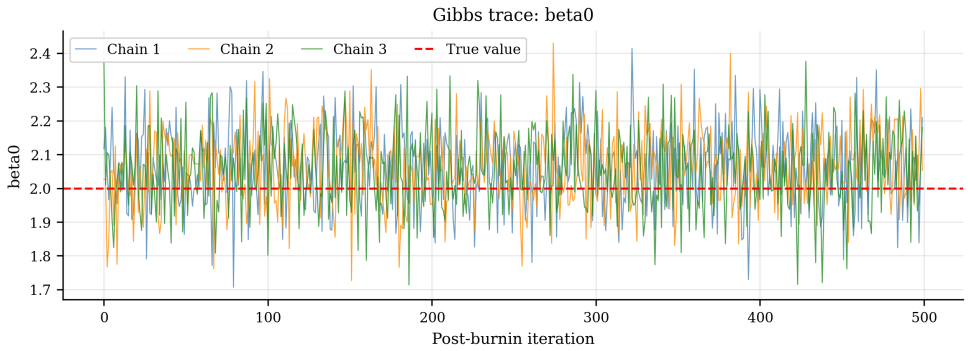

Figure 14: Gibbs trace plot for β_0 across 3 chains.

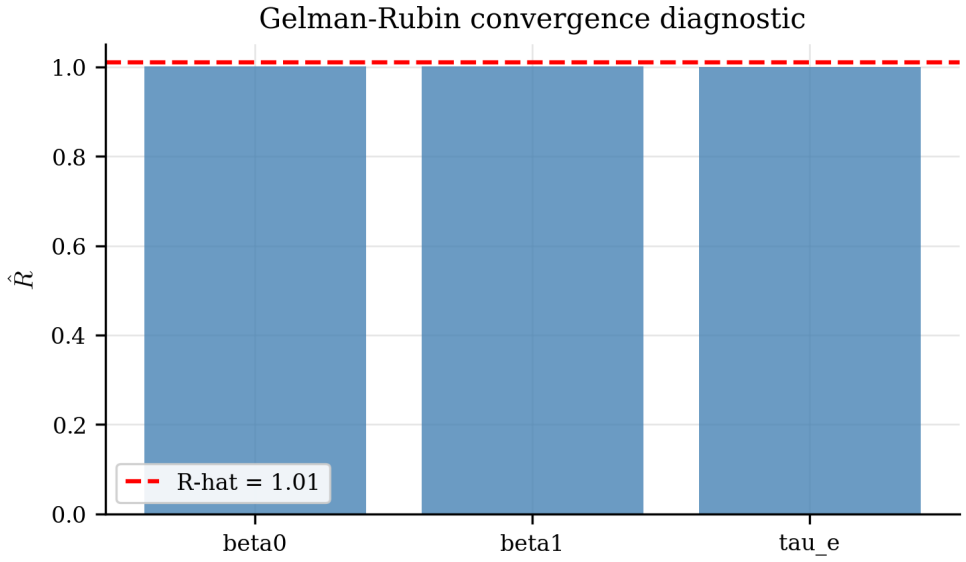

Figure 15: Gelman–Rubin $\hat{R}$ statistic for all parameters. All values are below the standard convergence threshold of 1.01.

8.2 Discussion

The linear test case is a favourable setting for mean-field VI. The posterior is unimodal and close to Gaussian, so the factorisation $q(\boldsymbol{\beta}, \tau_e) = q(\boldsymbol{\beta})\, q(\tau_e)$ does less damage here than it would in a more strongly coupled model. CAVI also has the advantage of using the conjugate structure directly. Its SD ratios of 0.98–0.98 are therefore close to one, even though the approximation still imposes independence between $\boldsymbol{\beta}$ and τ_e.
The gradient-based methods reveal two distinct failure modes:

- Newton's method and BFGS converge to local optima substantially worse than the CAVI solution, showing that generic optimisers without analytical gradients can fail even on a convex ELBO landscape.
- Gradient ascent with finite-difference gradients offers no speed advantage over Gibbs on small problems: the $O(2d)$ function evaluations per gradient step dominate the total cost at $d = 7$ unconstrained parameters.

For the broader study, I treat this result as a useful warning. When conjugacy is available, CAVI is the natural method and generic gradient methods should not be expected to improve on it. When conjugacy is unavailable, however, gradient-based ELBO optimisation becomes much more important. In that setting, analytical gradients or automatic differentiation are needed; finite-difference gradients are unlikely to scale well.

9 Conclusion

I used Bayesian linear regression as a checked setting for comparing CAVI, gradient ascent, Newton's method, and BFGS on the same variational inference problem. The formulation is an ELBO maximisation problem, but the numerical question is broader than maximisation alone. The main issue is whether a converged variational approximation also gives a credible account of posterior uncertainty. Cholesky and log-space reparameterisations keep the covariance and Gamma parameters valid during optimisation, while the Gibbs sampler and the analytical posterior provide references for posterior spread.

My key conclusion is that optimisation performance and uncertainty quality must be read separately. A method can make progress on the ELBO while still producing a posterior approximation that is too narrow. That distinction is the main lesson of this linear case study and the reason I report posterior standard-deviation ratios alongside convergence results.

References

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, *112*(518), 859–877. https://doi.org/10.1080/01621459.2017.1285773

Jaakkola, T. S., & Jordan, M. I. (2000). Bayesian parameter estimation via variational methods. *Statistics and Computing*, *10*(1), 25–37. https://doi.org/10.1023/A:1008932416310

Nocedal, J., & Wright, S. J. (2006). *Numerical optimization* (2nd ed.). Springer. https://doi.org/10.1007/978-0-387-40065-5

Polson, N. G., Scott, J. G., & Windle, J. (2013). Bayesian inference for logistic models using Pólya–Gamma latent variables. *Journal of the American Statistical Association*, *108*(504), 1339–1349. https://doi.org/10.1080/01621459.2013.829001

Wainwright, M. J., & Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, *1*(1–2), 1–305. https://www.cs.columbia.edu/~blei/fogm/2020F/readings/WainwrightJordan2008.pdf

Appendix A VB Posterior Variance Collapse

Why Variance Collapse Happens

Variational inference finds $q(\boldsymbol{\theta})$ by minimising the KL divergence from q to the true posterior $p(\boldsymbol{\theta} \mid \boldsymbol{y})$:

$$\mathrm{KL}[q\|p] = \int q(\boldsymbol{\theta}) \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta} \mid \boldsymbol{y})}\, d\boldsymbol{\theta} \tag{41}$$

This is the forward, or exclusive, KL divergence. Its asymmetry matters for this report. The approximation is penalised when it puts mass in places where the true posterior is small, but it is not penalised in the same way for missing parts of the posterior tail. The resulting approximation tends to stay near the high-density centre of the posterior and can be too narrow.
The mean-field factorisation adds a second restriction. By forcing $q(\boldsymbol{\beta}, \tau_e) = q(\boldsymbol{\beta})\, q(\tau_e)$, it removes any posterior correlation between the coefficients and the precision. In this linear model the effect is mild, but it is still visible in the standard-deviation ratios.

Observation

Comparison of the VB posterior against the Gibbs sampler confirms this: the mean-field VB approximation systematically underestimates posterior variance. The SD ratio ($\sigma_{\mathrm{VB}}/\sigma_{\mathrm{Gibbs}}$) is reported for each parameter and method in Section 8. Ratios below 1.0 confirm collapse; the magnitude of the shortfall motivates the entropy regularisation approach described above.

Three Approaches to Prevent Collapse

1. Informative Prior on τ_e

Set non-zero prior hyperparameters (α_e, γ_e) to place a lower bound on the posterior variance. A tight prior on τ_e prevents the precision from being driven too high, which in turn keeps $\boldsymbol{\Sigma}_\beta$ from collapsing.
Mechanism: The CAVI update for b_e is:

$$b_e \leftarrow \gamma_e + \frac{1}{2}\left[\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_\beta\|^2 + \mathrm{tr}(\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\Sigma}_\beta)\right] \tag{42}$$

With $\gamma_e > 0$, b_e has a minimum floor, bounding $\mathbb{E}[\tau_e] = a_e/b_e$ from above.
Limitation: Changes the model; requires prior elicitation.

2. Variance Lower Bound Constraint

After each CAVI update, clip $\boldsymbol{\Sigma}_\beta$ so its diagonal never falls below a minimum threshold $\sigma^2_{\min}$:

$$\Sigma_{\beta,jj} \leftarrow \max\left(\Sigma_{\beta,jj},\ \sigma^2_{\min}\right) \quad \forall j \tag{43}$$

This was already noted in the Constraints section as an optional minimum variance constraint. In code:

```
Sigma_diag = np.diag(self.Sigma_beta)
np.fill_diagonal(self.Sigma_beta,
    np.maximum(Sigma_diag, sigma_min))
```

Limitation: Ad hoc; does not arise from a principled objective modification.

3. Entropy Regularisation of the ELBO

Add a regularisation term proportional to the entropy $H[q]$ of the variational distribution to the objective:

$$\mathcal{L}_{\text{reg}}(\boldsymbol{\phi}) = \mathcal{L}(\boldsymbol{\phi}) + \lambda \cdot H[q(\boldsymbol{\beta})] \tag{44}$$

For $q(\boldsymbol{\beta}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta)$, the Gaussian entropy is:

$$H[q(\boldsymbol{\beta})] = \frac{1}{2}\log\det(2\pi e\,\boldsymbol{\Sigma}_\beta) = \frac{p}{2}(1+\log 2\pi) + \frac{1}{2}\log\det(\boldsymbol{\Sigma}_\beta) \tag{45}$$

The regularised gradient with respect to $\boldsymbol{\Sigma}_\beta$ gains an additional term:

$$\nabla_{\boldsymbol{\Sigma}_\beta}\mathcal{L}_{\text{reg}} = \nabla_{\boldsymbol{\Sigma}_\beta}\mathcal{L} + \frac{\lambda}{2}\boldsymbol{\Sigma}_\beta^{-1} \tag{46}$$

This directly rewards larger $\boldsymbol{\Sigma}_\beta$, counteracting the collapse tendency. The hyperparameter $\lambda > 0$ controls the strength of the correction.
Of the three options, this is the one used in the numerical comparison because it modifies the objective directly and can be handled by the same gradient-based methods already implemented for this report.

Selected Approach for This Paper

Entropy regularisation (Approach 3) is implemented and compared against the unregularised ELBO. The regularisation parameter λ is treated as a hyperparameter tuned by comparing posterior interval coverage against the Gibbs reference. Results are presented as a secondary comparison alongside the main method comparison in Section 8.

Appendix B Cholesky Reparameterisation: Structural Constraint Satisfaction

Why Reparameterisation?

The gradient-based optimisers require a parameter vector they can update freely, but the variational parameters are not all free. In particular, $\boldsymbol{\Sigma}_\beta$ must remain positive definite and a_e, b_e must remain positive. Four possible strategies are:

1. **Ignore them** — the algorithm crashes when $\log|\boldsymbol{\Sigma}_\beta|$ becomes undefined or $a_e \leq 0$.
2. **Add a penalty term** — introduces a tuning parameter and ill-conditioning as $\rho \to \infty$.
3. **Project back to the feasible set** after each step — costs $O(p^3)$ per iteration and is non-differentiable at the boundary, breaking quasi-Newton methods such as BFGS.
4. **Reparameterise** so the constraints are encoded structurally and cannot be violated.

This report uses reparameterisation throughout. It avoids penalty tuning, avoids projection steps, and gives the optimisers a smooth unconstrained parameter vector.

Diagonal Positivity: Three Options

Once $\boldsymbol{\Sigma}_\beta = \boldsymbol{L}\boldsymbol{L}^T$ is adopted, a residual constraint remains: the diagonal entries L_{ii} must be strictly positive for $\boldsymbol{\Sigma}_\beta$ to be positive definite rather than merely positive semidefinite. Three options exist:

1. **Ignore the diagonal constraint** — positivity can still be violated, recovering the original problem.

2. **Softplus**: $L_{ii} = \log(1 + e^{u_i})$ — valid but less standard and numerically less clean.

3. **Log-space**: $L_{ii} = e^{\tilde{\ell}_{ii}}$, keeping $\tilde{\ell}_{ii} \in \mathbb{R}$ fully unconstrained — used here.

The parameter count is unchanged at $p(p+1)/2$ free entries: p log-diagonal entries $\tilde{\ell}_{ii}$ and $p(p-1)/2$ unconstrained off-diagonal entries ℓ_{ij}, $i > j$.

The Problem with Direct Optimisation

The covariance matrix $\boldsymbol{\Sigma}_\beta$ must satisfy $\boldsymbol{\Sigma}_\beta \succ 0$ (symmetric positive definite) at every iteration. If the optimiser updates $\boldsymbol{\Sigma}_\beta$ directly, a gradient step of the form

$$\boldsymbol{\Sigma}_\beta^{(t+1)} = \boldsymbol{\Sigma}_\beta^{(t)} + \alpha_t \nabla_{\boldsymbol{\Sigma}_\beta} \mathcal{L} \tag{47}$$

can produce a matrix that is *not* positive definite. Standard gradient-based methods operate on unconstrained Euclidean space and have no built-in mechanism to respect the positive-definiteness constraint — the constraint is simply violated as a by-product of the update, leading to numerical failure (negative variances, failed Cholesky in later steps, NaN in the ELBO).

The Cholesky Solution: Structural Guarantee

Cholesky decomposition eliminates the constraint by parameterising $\boldsymbol{\Sigma}_\beta$ in terms of a lower-triangular factor $\boldsymbol{L}$:

$$\boldsymbol{\Sigma}_\beta = \boldsymbol{L}\boldsymbol{L}^T, \qquad L_{ii} > 0 \quad \forall i \tag{48}$$

The useful property is simple: any lower-triangular matrix $\boldsymbol{L}$ with strictly positive diagonal entries produces a valid symmetric positive definite matrix:

- **Symmetry:** $(\boldsymbol{L}\boldsymbol{L}^T)^T = (\boldsymbol{L}^T)^T \boldsymbol{L}^T = \boldsymbol{L}\boldsymbol{L}^T$ — automatically symmetric.
- **Positive definiteness:** For any non-zero $\boldsymbol{v} \in \mathbb{R}^p$,

$$\boldsymbol{v}^T \boldsymbol{L}\boldsymbol{L}^T \boldsymbol{v} = \|\boldsymbol{L}^T \boldsymbol{v}\|^2 > 0 \tag{49}$$

  provided $\boldsymbol{L}^T$ has full column rank, which holds whenever all $L_{ii} > 0$ (since $\boldsymbol{L}$ is lower triangular, its rank equals the number of non-zero diagonal entries).

Updating $\boldsymbol{L}$ therefore avoids the main failure of direct covariance updates. The constraint is not repaired after the fact by a penalty or projection; it is built into the parameterisation.

Unconstrained Optimisation Over $\boldsymbol{\ell}$

The remaining difficulty is that $L_{ii} > 0$ is still an inequality constraint. This is resolved by log-parameterising the diagonal:

$$L_{ii} = e^{\ell_{ii}}, \quad \ell_{ii} \in \mathbb{R} \qquad \text{(diagonal)} \tag{50}$$

$$L_{ij} = \ell_{ij}, \quad \ell_{ij} \in \mathbb{R} \qquad \text{(lower off-diagonal)} \tag{51}$$

Since $e^{\ell_{ii}} > 0$ for all $\ell_{ii} \in \mathbb{R}$, the optimisation is now over a *fully unconstrained* parameter vector $\boldsymbol{\ell} \in \mathbb{R}^{p(p+1)/2}$. No projection, no clipping, no penalty — any point in $\mathbb{R}^{p(p+1)/2}$ maps to a valid $\boldsymbol{L}$, and hence a valid $\boldsymbol{\Sigma}_\beta = \boldsymbol{L}\boldsymbol{L}^T$.

Analogy with Log-Space Parameterisation

This is the same idea used for the Gamma parameters $a_e, b_e > 0$: instead of constraining $a_e > 0$ directly, we optimise over unconstrained $\eta_a = \log a_e \in \mathbb{R}$ and recover $a_e = e^{\eta_a}$, which is always positive. Cholesky + log-diagonal does the same thing for the cone of symmetric positive definite matrices: it provides a smooth bijection between unconstrained Euclidean space and the constraint set, so standard optimisation machinery applies without modification.

Appendix C Python Implementation Structure

Computational Complexity

- Gradient computation: $O(np^2)$ per evaluation (dominated by matrix operations)
- Hessian computation: $O(np^2 + d^3)$ for Newton's method
- Must handle special functions: digamma $\psi(\cdot)$, trigamma $\psi'(\cdot)$ for the Gamma distribution terms in the ELBO

The implementation follows a top-down sequential process across five stages. Each stage depends on the outputs of the previous one and saves its results to parquet before proceeding, so that any stage can be re-run independently without repeating expensive computation.

```
# Stage 1: Data generation
#    - generate_linear_case  ->  data/linear_data.parquet
#                                figs/linear_data_scatter.png

# Stage 2: ELBO evaluation
#    - unpack unconstrained parameters (mu_beta, L_chol, log_ae, log_be)
#    - reconstruct Sigma_beta via Cholesky: Sigma = L @ L.T
#    - reconstruct a_e, b_e via exp transforms
#    - evaluate standard or entropy-regularised ELBO
#    - verify analytical gradient against finite differences (<= 1e-5)
#                                figs/linear_elbo_gradient_check.png
#                                figs/linear_elbo_landscape.png
#                                results/tables/linear_gradient_check.tex

# Stage 3: Optimisation methods
#    - run_cavi
#    - run_gradient_ascent    (step-size history recorded)
#    - run_newton             (Hessian condition number tracked)
#    - run_bfgs               (via scipy.optimize.minimize)
#                            ->  data/linear_opt.parquet
#                                figs/linear_elbo_convergence.png
#                                figs/linear_gradient_stepsize.png
#                                figs/linear_newton_condition.png

# Stage 4: Reference sampler
#    - run_gibbs_sampler      (3 chains, configurable burn-in)
#                            ->  data/linear_gibbs.parquet
#                                figs/linear_gibbs_trace_{beta0,beta1,taue}.png
#                                figs/linear_gibbs_acf_{beta0,beta1,taue}.png
```

```
#                                   figs/linear_gibbs_rhat.png

# Stage 5: Diagnostics and comparison
#    - final ELBO per method
#    - number of iterations / runtime / convergence status
#    - posterior mean error:  |mu_VB - mu_Gibbs|
#    - posterior SD ratio:    sigma_VB / sigma_Gibbs  (variance collapse)
#    - VB vs Gibbs overlay:   figs/linear_comparison_{beta0,beta1,taue}.png
#    - VB vs Gibbs density:   figs/linear_vb_gibbs_{beta0,beta1,taue}.png
#                            ->  figs/linear_sd_ratios.png
#                                figs/linear_mean_errors.png
#                                results/tables/linear_performance.tex
#                                results/tables/linear_posterior_summary.tex
```

The implementation uses three files. `linear_run.ipynb` is the development environment: functions are written and debugged interactively in the notebook first, then migrated to the shared `.py` files once they are working correctly. This keeps the `.py` files as stable, tested code while the notebook serves as the exploratory workspace.

Notebook: `linear_run.ipynb`

New functions are developed and debugged here cell by cell before being committed to the `.py` files. The notebook imports from `vb_algorithms_py` and `vb_utils_py` once a function has been migrated, so it gradually thins to a pure runner. Each of the five stage blocks saves its outputs to parquet before proceeding, so any stage can be re-run independently without repeating earlier computation.

1. **Data generation** — generates and saves `linear_data.parquet`; produces `linear_data_scatter.png` (1 figure).
2. **ELBO evaluation** — verifies analytical gradient against finite differences at the CAVI optimum; produces gradient-check bar and ELBO landscape slice (2 figures). Must pass before any optimiser is run.
3. **Optimisation methods** — runs CAVI, gradient ascent, Newton, and BFGS; saves `linear_opt.parquet`; produces ELBO convergence curves, step-size history, and Hessian condition-number history (3 figures).
4. **Reference sampler** — runs Gibbs sampler (3 chains); saves `linear_gibbs.parquet`; produces trace plots, ACF plots, $\hat{R}$ bar, and per-component distribution plots (7 figures).
5. **Diagnostics and comparison** — loads all parquet files; produces VB/Gibbs overlay plots, SD-ratio bar, and posterior mean-error bar; renders the three `tblr` tables to `results/tables/` and displays inline PNG previews (8 figures, 3 tables).

Total outputs for the linear case: **21 figures** saved to `figs/` and **3 tables** saved to `results/tables/`.

File: `vb_algorithms_py.py`

Contains all algorithm logic. Functions are migrated here from the notebook once verified.

- **ELBO evaluation** — unpacks unconstrained parameter vector, reconstructs $\mathbf{\Sigma}_\beta$ via Cholesky lower-triangular factor, reconstructs a_e and b_e via exp transforms, evaluates standard or entropy-regularised ELBO.

- **Analytical gradient** — computes $\nabla_{\boldsymbol{\xi}}\mathcal{L}$ with respect to all unconstrained parameters; verified against finite differences to relative error $\leq 10^{-5}$.
- **Hessian** — computes the Hessian of the ELBO for Newton's method; condition number tracked per iteration.
- **CAVI baseline** — coordinate ascent updates for $q(\boldsymbol{\beta})$ and $q(\tau_e)$.
- **Gradient ascent** — fixed or line-search step size; step-size history recorded.
- **Newton's method** — Hessian-based updates; falls back on gradient step if Hessian indefinite.
- **BFGS** — quasi-Newton method via `scipy.optimize.minimize`.
- **Gibbs sampler** — blocked sampler for $\boldsymbol{\beta} \mid \tau_e, \boldsymbol{y}$ and $\tau_e \mid \boldsymbol{\beta}, \boldsymbol{y}$; multiple chains with configurable burn-in.

File: `vb_utils_py.py`

Contains data generation, plotting, parquet I/O, and table rendering utilities. Functions are migrated here from the notebook once verified.

- **Data generation** — `generate_linear_case`: generates design matrix $\boldsymbol{X}$, response $\boldsymbol{y}$, and true parameter values for the linear Bayesian regression test case.
- **Parquet I/O** — `save_results_parquet`: writes numerical results to `results/data/*.parquet` as the source of truth; `load_results_parquet`: reloads without rerunning computation.
- **Table rendering** — `render_table_tex`: reads a polars DataFrame, formats values, writes a `tabularray` (`tblr`) snippet to `results/tables/*.tex` for `\input{}` inclusion in the report; also compiles a standalone preview PNG for inline notebook display.
- **Plot helpers** — `save_fig`: saves to `figs/` with consistent style; helpers for scatter plots, KDE overlays, bar charts, ACF, trace plots, ELBO curves, step-size histories, Hessian condition-number histories.

Appendix D Closed-Form Entropy Derivations

The ELBO decomposes as $\mathcal{L} = \mathbb{E}_q[\log p(\boldsymbol{y}, \boldsymbol{\theta})] + H[q]$, where the entropy $H[q] = -\mathbb{E}_q[\log q(\boldsymbol{\theta})]$ factors under the mean-field assumption into one term per variational factor. Both factors in this model belong to the exponential family, so each entropy has a closed-form expression derived below.

Gaussian Entropy: $H[q(\boldsymbol{\beta})]$

The general p-dimensional multivariate Gaussian density for a random vector $\boldsymbol{x} \in \mathbb{R}^p$ with mean $\boldsymbol{\mu}$ and covariance Σ is

$$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\Big(-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\Big). \tag{52}$$

The variational factor for the regression coefficients is $q(\boldsymbol{\beta}) = \mathcal{N}(\boldsymbol{\mu}_\beta, \Sigma_\beta)$, so substituting $\boldsymbol{x} = \boldsymbol{\beta}$, $\boldsymbol{\mu} = \boldsymbol{\mu}_\beta$, $\Sigma = \Sigma_\beta$:

$$q(\boldsymbol{\beta}) = \frac{1}{(2\pi)^{p/2}|\Sigma_\beta|^{1/2}} \exp\Big(-\tfrac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)^T\Sigma_\beta^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)\Big). \tag{53}$$

Taking the natural logarithm of both sides:

$$\begin{aligned}\log q(\boldsymbol{\beta}) &= \log \frac{1}{(2\pi)^{p/2}|\Sigma_\beta|^{1/2}} - \tfrac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)^T\Sigma_\beta^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta) \\ &= -\tfrac{p}{2}\log(2\pi) - \tfrac{1}{2}\log|\Sigma_\beta| - \tfrac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)^T\Sigma_\beta^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta). \end{aligned} \tag{54}$$

Taking the expectation under $q(\boldsymbol{\beta})$, the first two terms are constants and pass through unchanged. For the quadratic term, the scalar $\boldsymbol{v}^T A\boldsymbol{v} = \operatorname{tr}(A\boldsymbol{v}\boldsymbol{v}^T)$ for any vector $\boldsymbol{v}$ and matrix A, so

$$\begin{aligned}\mathbb{E}_q\big[(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)^T\Sigma_\beta^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)\big] &= \mathbb{E}_q\big[\operatorname{tr}\big(\Sigma_\beta^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)^T\big)\big] \\ &= \operatorname{tr}\big(\Sigma_\beta^{-1}\,\mathbb{E}_q\big[(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)^T\big]\big) \\ &= \operatorname{tr}(\Sigma_\beta^{-1}\Sigma_\beta) \\ &= \operatorname{tr}(I_p) = p, \end{aligned} \tag{55}$$

where the second equality uses linearity of expectation inside the trace, and the third uses the definition of Σ_β as the covariance of $q(\boldsymbol{\beta})$. Negating the full expectation gives the entropy:

$$\begin{aligned}H[q(\boldsymbol{\beta})] &= -\mathbb{E}_q[\log q(\boldsymbol{\beta})] \\ &= \tfrac{p}{2}\log(2\pi) + \tfrac{1}{2}\log|\Sigma_\beta| + \tfrac{1}{2}p \\ &= \tfrac{1}{2}\log|\Sigma_\beta| + \tfrac{p}{2}\big(1+\log 2\pi\big). \end{aligned} \tag{56}$$

In code this is computed as

```
sign, logdet = np.linalg.slogdet(Sigma)          # log|Sigma|
neg_lq_beta  = 0.5*logdet + 0.5*p*(1 + np.log(2*np.pi))
```

where `slogdet` returns the sign and log-magnitude of the determinant stably for near-singular matrices.

Gamma Entropy: $H[q(\tau_e)]$

The variational factor for the noise precision is $q(\tau_e) = \text{Gamma}(a_e, b_e)$ under the rate parameterisation, so the density is $q(\tau_e) = b_e^{a_e}\,\tau_e^{a_e-1}\,e^{-b_e\tau_e}/\Gamma(a_e)$ and its log is

$$\log q(\tau_e) = a_e\log b_e - \log\Gamma(a_e) + (a_e-1)\log\tau_e - b_e\tau_e. \tag{57}$$

The required expectations under $q(\tau_e)$ are standard results:

$$\mathbb{E}[\tau_e] = \frac{a_e}{b_e}, \qquad \mathbb{E}[\log\tau_e] = \psi(a_e) - \log b_e, \tag{58}$$

where $\psi(\cdot)$ is the digamma function. Substituting:

$$\begin{aligned}\mathbb{E}_q[\log q(\tau_e)] &= a_e\log b_e - \log\Gamma(a_e) + (a_e-1)\big(\psi(a_e)-\log b_e\big) - b_e\cdot\frac{a_e}{b_e} \\ &= a_e\log b_e - \log\Gamma(a_e) + (a_e-1)\psi(a_e) - (a_e-1)\log b_e - a_e \\ &= \log b_e - \log\Gamma(a_e) + (a_e-1)\psi(a_e) - a_e. \end{aligned} \tag{59}$$

Negating to obtain the entropy:

$$H[q(\tau_e)] = a_e - \log b_e + \ln\Gamma(a_e) + (1-a_e)\psi(a_e). \tag{60}$$

In code:

```
neg_lq_tau = a_e - np.log(b_e) + gammaln(a_e) + (1 - a_e)*digamma(a_e)
```

where `gammaln` and `digamma` are imported from `scipy.special`.