VI-Unified f(.) Functions Documentation

0.1 exact_linear_posterior()

Purpose: Computes the analytical conjugate posterior for linear regression under Normal-Inverse-Gamma prior.

Arguments:

- `X`: Design matrix ($N \times K$)
- `y`: Response vector ($N \times 1$)
- `prior_mu`: Prior mean for $\boldsymbol{\beta}$ (default: $\mathbf{0}_K$)
- `prior_sigma`: Prior standard deviation for $\boldsymbol{\beta}$ (default: 10)
- `prior_a`: Inverse-Gamma shape parameter for σ^2 (default: 2)
- `prior_b`: Inverse-Gamma rate parameter for σ^2 (default: 2)

Calculations:

Prior: $\boldsymbol{\beta} \sim N(\boldsymbol{\mu}_0, \sigma^2\boldsymbol{\Lambda}_0^{-1})$, $\sigma^2 \sim \mathrm{IG}(a_0, b_0)$

Where $\boldsymbol{\Lambda}_0 = \mathrm{diag}(K)/\sigma^2_{\mathrm{prior}}$

Posterior parameters:

$$\begin{aligned}
\boldsymbol{\Sigma}_n &= (\boldsymbol{\Lambda}_0 + \mathbf{X}^\top\mathbf{X})^{-1} \\
\boldsymbol{\mu}_n &= \boldsymbol{\Sigma}_n(\boldsymbol{\Lambda}_0\boldsymbol{\mu}_0 + \mathbf{X}^\top\mathbf{y}) \\
a_n &= a_0 + N/2 \\
b_n &= b_0 + \frac{1}{2}\left(\mathbf{y}^\top\mathbf{y} - \boldsymbol{\mu}_n^\top\boldsymbol{\Sigma}_n^{-1}\boldsymbol{\mu}_n\right)
\end{aligned}$$

Returns:

- `mu_n`: Posterior mean $\boldsymbol{\mu}_n$ for $\boldsymbol{\beta}$ (stored as `mu_n`)
- `Sigma_n`: Posterior covariance $\boldsymbol{\Sigma}_n$ (conditional on σ^2, stored as `Sigma_n`)
- `a_n`: Posterior Inverse-Gamma shape a_n (stored as `a_n`)
- `b_n`: Posterior Inverse-Gamma rate b_n (stored as `b_n`)
- `df`: Degrees of freedom $2a_n$
- `scale_matrix`: $(b_n/a_n)\boldsymbol{\Sigma}_n$ for multivariate-t marginal

0.2 sample_exact_posterior()

Purpose: Draws samples from the Normal-Inverse-Gamma posterior using conditional sampling.

Arguments:

- `exact_post`: List returned by `exact_linear_posterior()`
- `n_samples`: Number of posterior samples (default: 2000)

Calculations:

Two-stage sampling:

1. Draw $\sigma^2 \sim \mathrm{IG}(a_n, b_n)$ via $\sigma^2 = 1/\mathrm{Gamma}(a_n, b_n)$
2. Draw $\boldsymbol{\beta} \mid \sigma^2 \sim N(\boldsymbol{\mu}_n, \sigma^2 \boldsymbol{\Sigma}_n)$

Repeat for `n_samples` iterations.

Returns:

- Matrix $(n_{\text{samples}} \times K + 1)$ with columns: $[\beta_1, \ldots, \beta_K, \sigma]$

0.3 run_laplace()

Purpose: Laplace approximation via maximum a posteriori (MAP) estimation and inverse Hessian.

Arguments:

- `X`: Design matrix for fixed effects ($N \times p$)
- `Z`: Design matrix for random effects ($N \times q$)
- `y`: Response vector ($N \times 1$)
- `p`: Number of fixed effects
- `q`: Number of random effects
- `n`: Number of observations
- `alpha_e`, `gamma_e`: Gamma prior parameters for τ_e
- `alpha_u`, `gamma_u`: Gamma prior parameters for τ_u
- `model_type`: `"M1"` (linear) or `"M3"` (hierarchical)

Calculations:

For M1: Optimise $\boldsymbol{\theta} = (\boldsymbol{\beta}, \log \tau_e)$

Negative log-posterior:

$$-\log p(\boldsymbol{\theta} \mid \mathbf{y}) = -\left[\underbrace{-\frac{N}{2}\log(2\pi) + \frac{N}{2}\log \tau_e - \frac{\tau_e}{2}\sum r_i^2}_{\text{likelihood}} \right.$$
$$\left. + \underbrace{(\alpha_e - 1)\log \tau_e - \gamma_e \tau_e}_{\text{prior for } \tau_e} + \underbrace{-\frac{1}{200}\|\boldsymbol{\beta}\|^2}_{\text{prior for } \boldsymbol{\beta}}\right]$$

Where $r_i = y_i - \mathbf{X}_i \boldsymbol{\beta}$

For M3: Optimise $\boldsymbol{\theta} = (\boldsymbol{\beta}, \mathbf{u}, \log \tau_e, \log \tau_u)$

Additional prior term for random effects:

$$-\frac{q}{2}\log(2\pi) + \frac{q}{2}\log \tau_u - \frac{\tau_u}{2}\mathbf{u}^\top \mathbf{K}^{-1}\mathbf{u}$$

Plus prior: $(\alpha_u - 1)\log \tau_u - \gamma_u \tau_u$

Use BFGS optimisation to find $\hat{\boldsymbol{\theta}}_{\text{MAP}}$, compute Hessian $\mathbf{H}$, approximate:

$$q(\boldsymbol{\theta}) \approx N(\hat{\boldsymbol{\theta}}_{\text{MAP}}, \mathbf{H}^{-1})$$

Returns:

- `theta_map`: MAP estimate $\hat{\boldsymbol{\theta}}_{\text{MAP}}$
- `cov_matrix`: Inverse Hessian $\mathbf{H}^{-1}$
- `model_type`, `p`, `q`: Metadata

0.4 run_gibbs_sampler()

Purpose: Full Gibbs sampler (MCMC) for linear or hierarchical model.

Arguments:

- `X`, `Z`, `y`, `p`, `q`, `n`: As in `run_laplace()`
- `alpha_e`, `gamma_e`, `alpha_u`, `gamma_u`: Prior hyperparameters
- `model_type`: `"M1"` or `"M3"`
- `n_iter`: Total MCMC iterations (default: 5000)
- `n_burnin`: Burn-in iterations (default: 1000)

Calculations:

For M1: Gibbs sampling for $(\boldsymbol{\beta}, \tau_e)$

Full conditionals:

$$\begin{aligned}
\boldsymbol{\beta} \mid \tau_e, \mathbf{y} &\sim N(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) \\
\boldsymbol{\Sigma}_\beta &= (\tau_e \mathbf{X}^\top \mathbf{X} + \mathbf{I}_p/100)^{-1} \\
\boldsymbol{\mu}_\beta &= \tau_e \boldsymbol{\Sigma}_\beta \mathbf{X}^\top \mathbf{y} \\
\tau_e \mid \boldsymbol{\beta}, \mathbf{y} &\sim \text{Gamma}(a_{\text{post}}, b_{\text{post}}) \\
a_{\text{post}} &= \alpha_e + N/2 \\
b_{\text{post}} &= \gamma_e + \frac{1}{2}\sum(y_i - \mathbf{X}_i\boldsymbol{\beta})^2
\end{aligned}$$

For M3: Gibbs sampling for $(\boldsymbol{\beta}, \mathbf{u}, \tau_e, \tau_u)$

Joint update for $(\boldsymbol{\beta}, \mathbf{u})$:

$$\begin{aligned}
\begin{bmatrix} \boldsymbol{\beta} \\ \mathbf{u} \end{bmatrix} \mid \tau_e, \tau_u, \mathbf{y} &\sim N(\boldsymbol{\mu}_{\beta u}, \boldsymbol{\Sigma}_{\beta u}) \\
\boldsymbol{\Sigma}_{\beta u} &= \left(\tau_e [\mathbf{X}|\mathbf{Z}]^\top [\mathbf{X}|\mathbf{Z}] + \tau_u \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}^{-1} \end{bmatrix} + \mathbf{I}_{p+q}/100 \right)^{-1} \\
\boldsymbol{\mu}_{\beta u} &= \tau_e \boldsymbol{\Sigma}_{\beta u} [\mathbf{X}|\mathbf{Z}]^\top \mathbf{y}
\end{aligned}$$

Update τ_e:

$$\tau_e \mid \boldsymbol{\beta}, \mathbf{u}, \mathbf{y} \sim \text{Gamma}(\alpha_e + N/2, \gamma_e + \frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\mathbf{u}\|^2)$$

Update τ_u:

$$\tau_u \mid \mathbf{u} \sim \text{Gamma}(\alpha_u + q/2, \gamma_u + \frac{1}{2}\mathbf{u}^\top \mathbf{K}^{-1} \mathbf{u})$$

Returns:

- Matrix of post-burn-in samples $(n_{\text{iter}} - n_{\text{burnin}}) \times d$
- M1: Columns $[\beta_0, \ldots, \beta_{p-1}, \tau_e]$
- M3: Columns $[\beta_0, \ldots, \beta_{p-1}, u_1, \ldots, u_q, \tau_e, \tau_u]$

0.5 plot_vb_posteriors()

Purpose: Creates 8-panel comparison plot showing VB, Exact, Laplace, and Gibbs posteriors.

Arguments:

- `mu_beta`: VB posterior mean for $(\boldsymbol{\beta}, \mathbf{u})$
- `Sigma_betau`: VB posterior covariance
- `mu_beta_exact`: Exact posterior mean for $\boldsymbol{\beta}$
- `Sigma_beta_exact`: Exact posterior covariance
- `laplace_result`: Output from `run_laplace()`
- `gibbs_samples`: Output from `run_gibbs_sampler()`
- `p`, `q`: Dimensions
- `beta_true`, `u_true`: True parameter values
- `tau_e_true`, `tau_u_true`: True precision values
- `E_tau_e`, `E_tau_u`: VB expectations
- `a_e_new`, `b_e_new`: VB Gamma parameters for τ_e
- `a_u_new`, `b_u_new`: VB Gamma parameters for τ_u
- `gibbs_tau_e`, `gibbs_tau_u`: Gibbs samples for precisions
- `run_gibbs`: Logical, whether Gibbs was run
- `model_type`: `"M1"` or `"M3"`

Calculations:

For each parameter $\theta_i \in \{\beta_0, \ldots, \beta_{p-1}, u_1, u_2, \tau_e, \tau_u\}$:

1. Construct marginal densities:
   - **VB**: $q(\theta_i) = N(\mu_i, \Sigma_{ii})$ for $\boldsymbol{\beta}, \mathbf{u}$; Gamma for τ_e, τ_u
   - **Exact**: $p(\theta_i \mid \mathbf{y})$ from analytical posterior
   - **Laplace**: $q(\theta_i) = N(\hat{\theta}_i, H_{ii}^{-1})$
   - **Gibbs**: Kernel density estimate from MCMC samples
2. Compute SD ratios:
$$\frac{\mathrm{SD}_{\mathrm{VB}}}{\mathrm{SD}_{\mathrm{Exact}}}, \quad \frac{\mathrm{SD}_{\mathrm{VB}}}{\mathrm{SD}_{\mathrm{Laplace}}}, \quad \frac{\mathrm{SD}_{\mathrm{VB}}}{\mathrm{SD}_{\mathrm{Gibbs}}}$$
3. Overlay densities with colour-coded lines and true value as vertical line

Returns:

- ggplot2 patchwork object with 8 panels (M3) or 5 panels (M1)
- Each panel shows 4 density overlays with SD ratio annotations

0.6 run_vb_algorithm()

Purpose: Mean-field variational Bayes with coordinate ascent for hierarchical models.

Arguments:

- `X`, `Z`, `y`: Data matrices
- `K`: Covariance structure for random effects (typically $\mathbf{I}_q$)
- `p`, `q`, `n`: Dimensions
- `alpha_e`, `gamma_e`, `alpha_u`, `gamma_u`: Prior hyperparameters
- `model_type`: `"M1"` or `"M3"`
- `max_iter`: Maximum iterations (default: 100)
- `tol`: Convergence tolerance (default: 10^{-5})

Calculations:

Mean-field factorisation:

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

Coordinate ascent updates:

1. Update $q(\boldsymbol{\beta}, \mathbf{u})$:

$$\begin{aligned} q(\boldsymbol{\beta}, \mathbf{u}) &\sim N(\boldsymbol{\mu}_{\beta u}, \boldsymbol{\Sigma}_{\beta u}) \\ \boldsymbol{\Sigma}_{\beta u} &= \left(E[\tau_e][\mathbf{X}|\mathbf{Z}]^\top [\mathbf{X}|\mathbf{Z}] + E[\tau_u] \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}^{-1} \end{bmatrix} + \mathbf{I}_{p+q}/100 \right)^{-1} \\ \boldsymbol{\mu}_{\beta u} &= E[\tau_e]\, \boldsymbol{\Sigma}_{\beta u}\, [\mathbf{X}|\mathbf{Z}]^\top \mathbf{y} \end{aligned}$$

Where `mu_betau` stores $\boldsymbol{\mu}_{\beta u}$, `Sigma_betau` stores $\boldsymbol{\Sigma}_{\beta u}$, and `E_tau_e`, `E_tau_u` store the expectations.

2. Update $q(\tau_e)$:

$$\begin{aligned} q(\tau_e) &\sim \text{Gamma}(a_e^{\text{new}}, b_e^{\text{new}}) \\ a_e^{\text{new}} &= \alpha_e + N/2 \\ b_e^{\text{new}} &= \gamma_e + \frac{1}{2}\left[\|\mathbf{y}\|^2 - 2\mathbf{y}^\top [\mathbf{X}|\mathbf{Z}]\boldsymbol{\mu}_{\beta u} + \text{tr}([\mathbf{X}|\mathbf{Z}]^\top [\mathbf{X}|\mathbf{Z}]\,(\boldsymbol{\Sigma}_{\beta u} + \boldsymbol{\mu}_{\beta u}\boldsymbol{\mu}_{\beta u}^\top)) \right] \end{aligned}$$

Where `a_e_new` stores a_e^{new}, `b_e_new` stores b_e^{new}, and `E_tau_e` $= a_e^{\text{new}}/b_e^{\text{new}}$.

3. Update $q(\tau_u)$:

$$\begin{aligned} q(\tau_u) &\sim \text{Gamma}(a_u^{\text{new}}, b_u^{\text{new}}) \\ a_u^{\text{new}} &= \alpha_u + q/2 \\ b_u^{\text{new}} &= \gamma_u + \frac{1}{2}\left[\underbrace{\boldsymbol{\mu}_u^\top \mathbf{K}^{-1} \boldsymbol{\mu}_u}_{\text{quad\_form}} + \underbrace{\text{tr}(\mathbf{K}^{-1}\boldsymbol{\Sigma}_{uu})}_{\text{trace\_u}} \right] \end{aligned}$$

Where `quad_form` $= \boldsymbol{\mu}_u^\top \mathbf{K}^{-1} \boldsymbol{\mu}_u$ measures the squared magnitude of $\boldsymbol{\mu}_u$ (the quadratic form), and `trace_u` $= \text{tr}(\mathbf{K}^{-1}\boldsymbol{\Sigma}_{uu})$ accounts for uncertainty in $\boldsymbol{\Sigma}_{uu}$ (the trace term).

The R code stores: `a_u_new` $= a_u^{\text{new}}$, `b_u_new` $= b_u^{\text{new}}$, `E_tau_u` $= a_u^{\text{new}}/b_u^{\text{new}}$.

Extracting random effects: $\boldsymbol{\mu}_u = \boldsymbol{\mu}_{\beta u}[(p+1):(p+q)]$ and $\boldsymbol{\Sigma}_{uu} = \boldsymbol{\Sigma}_{\beta u}[(p+1):(p+q),(p+1):(p+q)]$

ELBO:

$$\mathcal{L} = E_q[\log p(\mathbf{y}, \boldsymbol{\beta}, \mathbf{u}, \tau_e, \tau_u)] - E_q[\log q(\boldsymbol{\beta}, \mathbf{u}, \tau_e, \tau_u)]$$

Iterate until $|\mathcal{L}^{(t)} - \mathcal{L}^{(t-1)}| < \text{tol}$ or `max_iter` reached.

Returns:

- `mu_betau`: Posterior mean $\boldsymbol{\mu}_{\beta u}$
- `Sigma_betau`: Posterior covariance $\boldsymbol{\Sigma}_{\beta u}$
- `E_tau_e`, `E_tau_u`: a/b from Gamma posteriors
- `a_e_new`, `b_e_new`, `a_u_new`, `b_u_new`: Gamma parameters
- `elbo_history`: ELBO trajectory
- `E_tau_e_history`, `E_tau_u_history`: Parameter trajectories

0.7 plot_convergence()

Purpose: Diagnostic plots showing VB parameter convergence over iterations.

Arguments:

- `results`: Output from `run_vb_algorithm()`
- `scenario_name`: String label for plot title
- `tau_e_true`, `tau_u_true`: True parameter values
- `model_type`: `"M1"` or `"M3"`

Calculations:

Extract iteration histories:

- $E[\tau_e]^{(t)}$ for $t = 1, \ldots, T$
- $E[\tau_u]^{(t)}$ for $t = 1, \ldots, T$ (M3 only)

Create line plots with true values as horizontal reference lines.

Returns:

- ggplot2 patchwork object:
  - M1: Single plot for $E[\tau_e]$
  - M3: Two-panel plot for $E[\tau_e]$ and $E[\tau_u]$

0.8 plot_elbo()

Purpose: Plot Evidence Lower BOund (ELBO) trajectory for convergence diagnostics.

Arguments:

- `results`: Output from `run_vb_algorithm()`
- `scenario_name`: String label for plot title

Calculations:

Extract ELBO history: $\mathcal{L}^{(1)}, \mathcal{L}^{(2)}, \ldots, \mathcal{L}^{(T)}$

The ELBO should be:

- **Monotonically increasing**: Each coordinate ascent update improves (or maintains) the ELBO
- **Flattening**: Convergence indicated when slope approaches zero

Returns:

- ggplot2 line plot showing ELBO vs iteration number
- Helps identify: premature convergence, oscillations, or divergence