PML 2. Variational Inference

Probabilistic Machine Learning Reading Group

Pablo G. Arce

November 19, 2025

Institute of Mathematical Sciences (ICMAT-CSIC)

Contents

Contents

The Bayesian Inference Problem

- We model data x and latent variables z through a joint $p(x, z) = p(x \mid z)p(z)$.
- The goal is to infer the posterior:

$$p(z \mid x) = \frac{p(x \mid z)p(z)}{p(x)}.$$

- Examples:
  - **Regression:** predict outcomes with uncertainty intervals.
  - **Clustering:** infer mixture components and their probabilities.
  - **Neural networks:** estimate uncertainty in model parameters.

[Code S1]

The Bayesian Inference Problem

- We model data x and latent variables z through a joint $p(x, z) = p(x \mid z)p(z)$.
- The goal is to infer the posterior:

$$p(z \mid x) = \frac{p(x \mid z)p(z)}{p(x)}.$$

- This lets us:
  - Quantify uncertainty.
  - Compare models via $p(x)$.
  - Make predictions for new data.

Why Exact Inference is Hard

- The evidence

$$p(x) = \int p(x, z)\, dz$$

  is rarely tractable.
- Causes:
  - High-dimensional latent spaces.
  - Non-conjugate models.
  - Nonlinear likelihoods (e.g., neural nets).

Example: Intractable Posterior

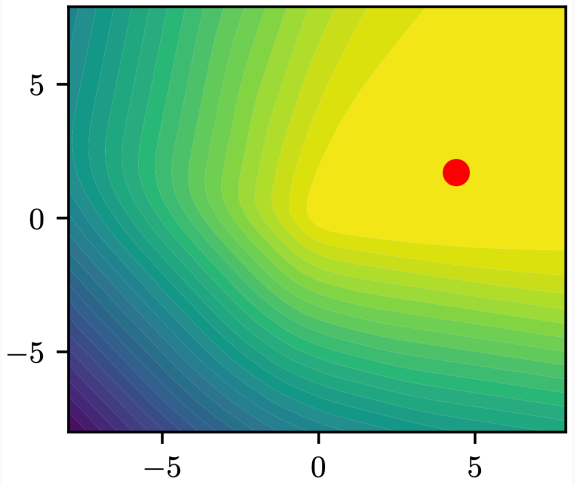
Bayesian logistic regression:

$$p(y_i = 1 \mid x_i, w) = \sigma(w^\top x_i), \quad p(w) = \mathcal{N}(0, I)$$

Posterior:

$$p(w \mid X, y) \propto p(w) \prod_i \sigma(w^\top x_i)^{y_i} (1 - \sigma(w^\top x_i))^{1-y_i}$$

- No closed form for $p(w \mid X, y)$.
- The integral for $p(X, y)$ has no analytic solution.
- Numerical integration infeasible for large w.



Example posterior for 2D weights — non-Gaussian and curved.

Key Idea of VI

- Introduce tractable distribution $q(z)$.
- Optimize it to approximate the true posterior.
- Turn inference:

$$p(z|x)$$

into optimization:

$$\arg\max_{q \in \mathcal{Q}} \mathcal{L}(q).$$

VI in Modern ML

- VAEs and deep generative models.
- Bayesian deep learning.
- Latent variable models: GMM, topic models.
- Probabilistic graphical models.

MCMC vs Variational Inference

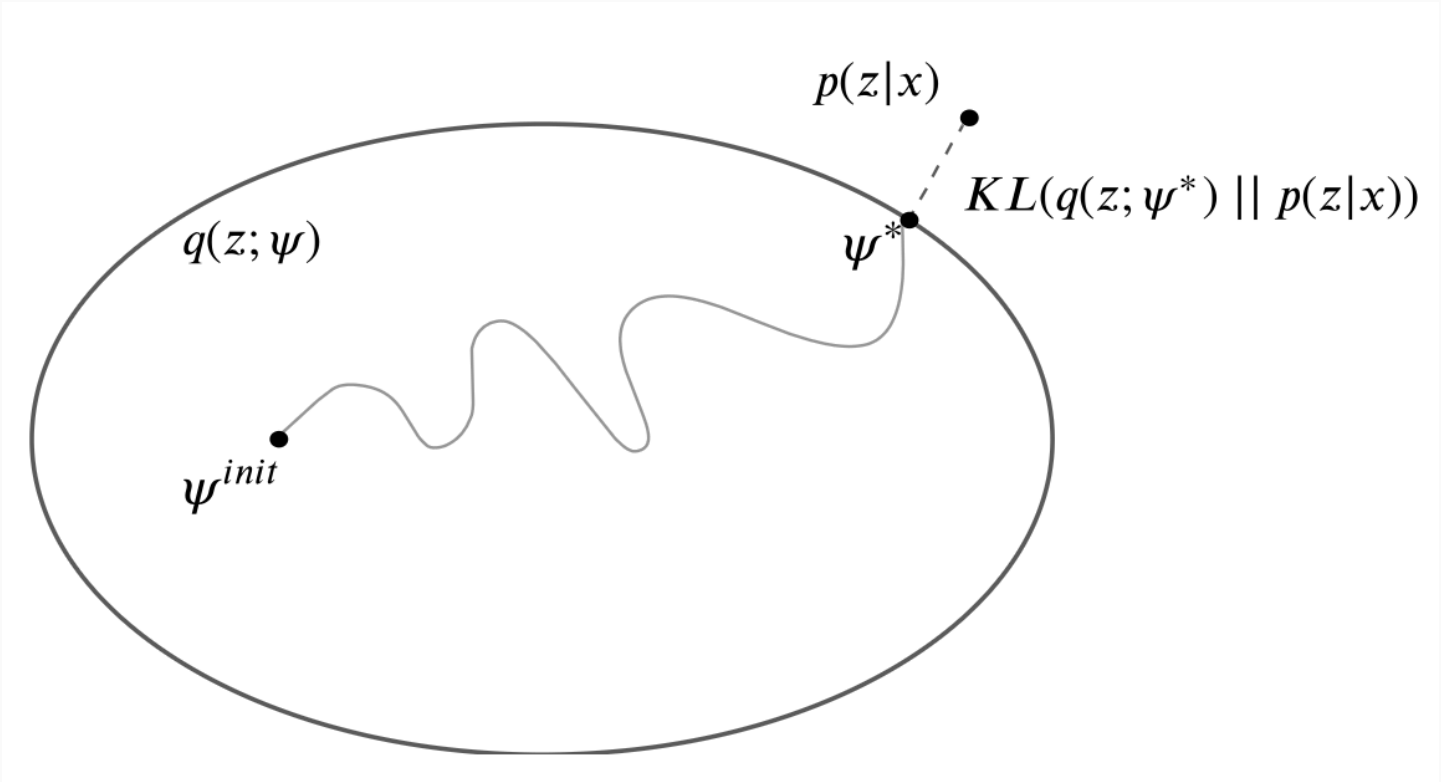
MCMC

- **Sample based.**
- Asymptotically exact.
- Slow for high dimension.
- Hard to scale.

VI

- **Approximation based.**
- Flexible families.
- Fast.
- Scales with SGD.

Visualization of Approximation



Contents

Key Idea of VI

$$\arg\max_{q \in \mathcal{Q}} \mathcal{L}(q, p(z|x))$$

Key Idea of VI

$$\arg\max_{q \in \mathcal{Q}} \mathcal{L}(q, p(z|x))$$

KL Divergence

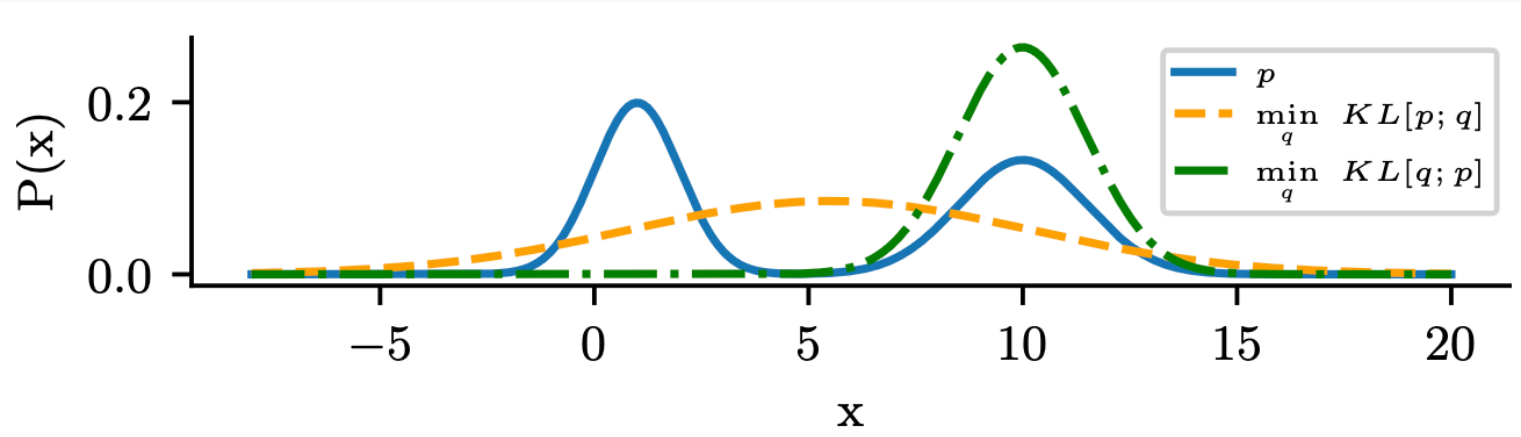
$$\mathrm{KL}(q\|p) = \mathbb{E}_q\left[\log \frac{q(z)}{p(z)}\right]$$

- Measures how different two distributions are.
- Always non-negative; zero only when $q = p$.
- Asymmetric:

$$\mathrm{KL}(q\|p) \neq \mathrm{KL}(p\|q).$$

Mode-Seeking vs. Mode-Covering

- Minimizing $\mathrm{KL}(q\|p)$:
  - Penalizes placing mass where p is low.
  - Encourages focusing on a single mode.
  - **Mode-seeking** behavior.
- Minimizing $\mathrm{KL}(p\|q)$:
  - Penalizes missing any region where p has mass.
  - Encourages spreading over all modes.
  - **Mode-covering** behavior.



[Code S2]

KL Divergence

- In Variational Inference, we minimize $\mathrm{KL}(q\|p)$ to find the closest tractable $q(z)$.

$$\mathrm{KL}(q\|p) = \mathbb{E}_q\left[\log q(z) - \log p(z|x)\right]$$

Why KL?

1. **Mathematical elegance:** Gives us the ELBO decomposition

$$\log p(x) = \mathsf{ELBO}(q) + \mathsf{KL}(q\|p)$$

2. **Information-theoretic foundation:** Natural measure of distribution difference
3. **Computational tractability:** Only requires evaluating $\log p(x, z)$ and sampling q
4. **Entropy connection:**

$$\mathsf{KL}(q\|p) = -H(q) - \mathbb{E}_q[\log p(z)]$$

5. **Practical success:** Works well for most applications

Alternative divergences exist but sacrifice at least one of these properties

Why Do We Use $\mathrm{KL}(q\|p)$ and Not $\mathrm{KL}(p\|q)$?

The ideal objective: Minimize

$$\mathrm{KL}(p\|q) = \mathbb{E}_{p(z|x)}[\log p(z|x) - \log q(z)]$$

- **Problem:** Requires sampling from $p(z|x)$ — but that's intractable!
- This is the whole problem we're trying to solve

What we can compute: $\mathrm{KL}(q\|p) = \mathbb{E}_{q(z)}[\log q(z) - \log p(z|x)]$

- Only requires sampling from $q(z)$ (which we design to be easy)
- Only requires evaluating $\log p(x, z)$ (the joint, which we have)

We use $\mathrm{KL}(q\|p)$ because it's **computable**,
not because it's theoretically optimal

Can We Do Better?

Alternative objectives that avoid KL$(q\|\|p)$:

1. **Expectation Propagation (EP)**
   - Minimizes KL$(p\|\|q)$ locally
   - Mode-covering, but more complex
2. **α-divergences**
   - Interpolate between KL$(q\|\|p)$ and KL$(p\|\|q)$
   - $\alpha = 1$: forward KL, $\alpha = 0$: reverse KL
3. **Mixture variational families**
   - $q(z) = \sum_k \pi_k q_k(z)$ can capture multiple modes

The Bottom Line

Why KL$(q\|p)$?

1. It's mathematically equivalent to maximizing the ELBO
2. It only requires evaluating $\log p(x, z)$ and sampling from q
3. It's computationally tractable for complex models

The price we pay:

- Mode-seeking behavior
- Underestimation of uncertainty
- May miss important regions of posterior

VI trades off **computational feasibility**
for **approximation quality**

ELBO Derivation

$$\begin{aligned}\mathrm{KL}(q\|p) &= \mathbb{E}_q\left[\log q(z) - \log p(z|x)\right] \\ &= \mathbb{E}_q\left[\log q(z) - \log p(x|z) - \log p(z) + \log p(x)\right]\end{aligned}$$

ELBO Derivation

$$\begin{aligned}\mathrm{KL}(q\|p) &= \mathbb{E}_q\left[\log q(z) - \log p(z|x)\right] \\ &= \mathbb{E}_q\left[\log q(z) - \log p(x|z) - \log p(z) + \log p(x)\right]\end{aligned}$$

and using the non-negativity of the KL divergence

$$\begin{aligned}\log p(x) &= \mathrm{KL}(q\|p) + \mathbb{E}_q\left[\log p(x|z) + \log p(z) - \log q(z)\right] \\ &\geq \mathbb{E}_q\left[\log p(x|z) + \log p(z) - \log q(z)\right] = ELBO(q)\end{aligned}$$

ELBO Derivation

We have:

$$\log p(x) = ELBO(q) + \mathrm{KL}(q||p) = \text{constant}.$$

- Maximize ELBO to approximate posterior.
- Equivalent to minimizing KL divergence.

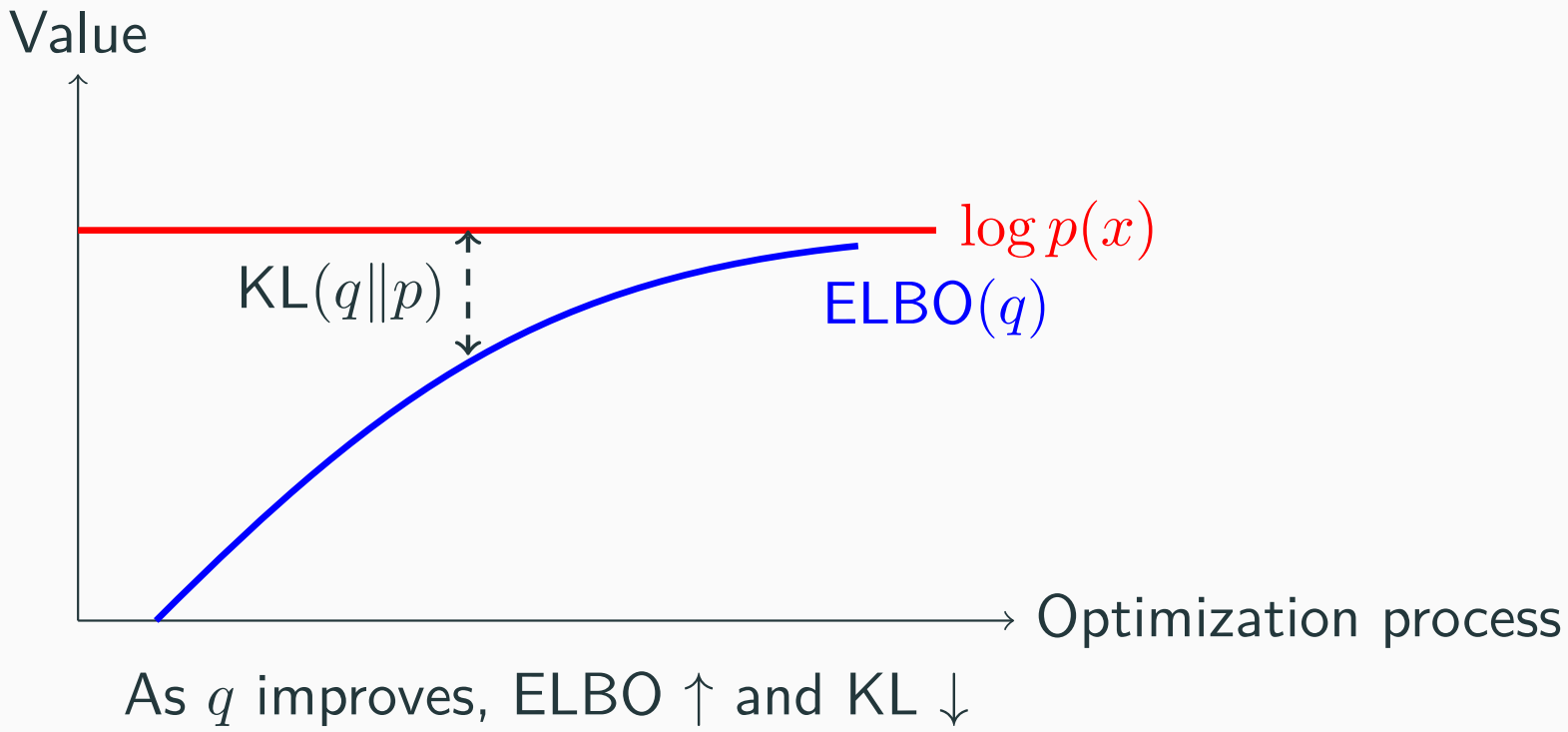
ELBO as Optimization Objective

$$ELBO(q) = \mathbb{E}_q \left[\log p(x|z) + \log p(z) - \log q(z)\right]$$

ELBO balances:

- Fit to data: $\mathbb{E}_q[\log p(x, z)]$
- Regularization: entropy of q

ELBO Intuition



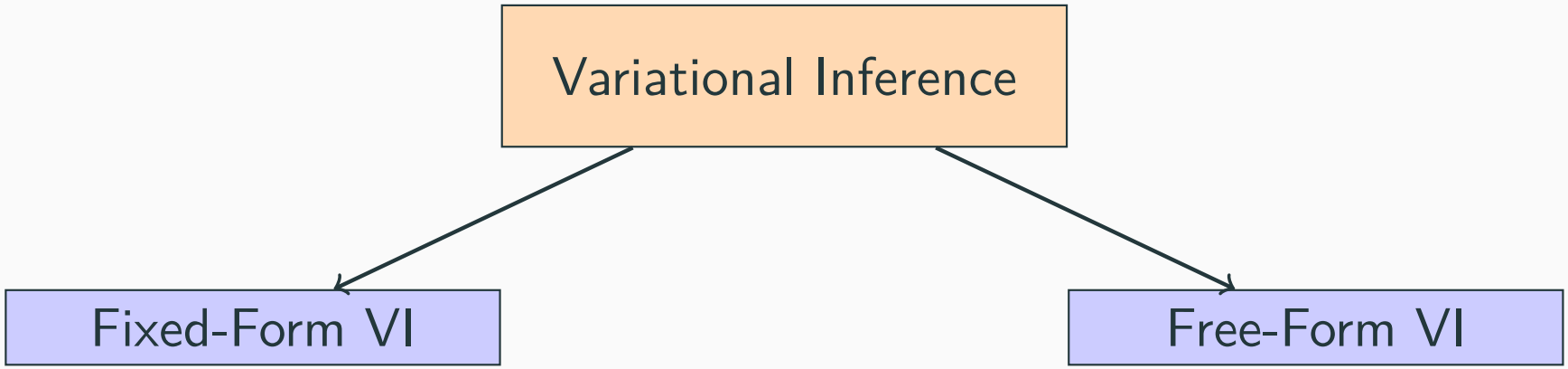
Key insight: Maximizing ELBO pushes q closer to $p(z|x)$

[Code S3]

Contents

Contents

Two Approaches to Variational Inference



Today's roadmap:

- Fixed-form: gradient-based optimization
- Free-form: coordinate ascent (CAVI)
- Scaling approaches with stochasticity (SVI)

Fixed-Form vs Free-Form VI

Fixed-Form VI

Choose the family first

$$q_\psi(z) \in \mathcal{Q}$$

Examples:

- Gaussian: $q(z) = \mathcal{N}(\mu, \Sigma)$
- Mean-field: $q(z) = \prod_i q_i(z_i)$
- Normalizing flows

Then optimize:

$$\psi^* = \arg\max_\psi \mathcal{L}(q_\psi)$$

Free-Form VI

Derive the optimal form

Calculus of variations:

$$\log q^*(z_i) = \mathbb{E}_{q_{\neg i}}[\log p(x, z)] + C$$

- Don't choose q parametrically
- Find best q within constraints
- Requires conjugacy for tractability

Contents

Fixed-Form VI: The Setup

Goal: Find the best approximation in family $\mathcal{Q}$

$$\psi^* = \arg\max_{\psi} \mathcal{L}(q_\psi) \quad \text{where} \quad \mathcal{L}(q) = \mathbb{E}_q[\log p(x, z)] - \mathbb{E}_q[\log q(z)]$$

Key insight: Turn inference into optimization

- Pick any differentiable family $q_\psi(z)$
- Adjust ψ via gradient ascent
- Maximizing ELBO $\Leftrightarrow$ Minimizing $\mathrm{KL}(q\|p)$

Challenge: How do we compute $\nabla_\psi \mathcal{L}(q_\psi)$?

The Gradient Challenge

Want to compute:

$$\nabla_\psi \mathcal{L}(q_\psi) = \nabla_\psi \mathbb{E}_{q_\psi}[\log p(x,z) - \log q_\psi(z)]$$

Problem: Gradient operator doesn't go inside expectation easily

$$z \sim q_\psi(z) \quad \Rightarrow \quad \text{sampling is not differentiable!}$$

Two solutions:

1. **Score function estimator** — General but high variance
2. **Reparameterization trick** — Low variance when applicable

Solution 1: Score Function Estimator

Key identity: (also called REINFORCE, likelihood-ratio)

$$\nabla_\psi \mathbb{E}_{q_\psi}[f(z)] = \mathbb{E}_{q_\psi}[f(z)\, \nabla_\psi \log q_\psi(z)]$$

Pros:

- Black-box: works for any q_ψ
- No constraints on the distribution

Cons:

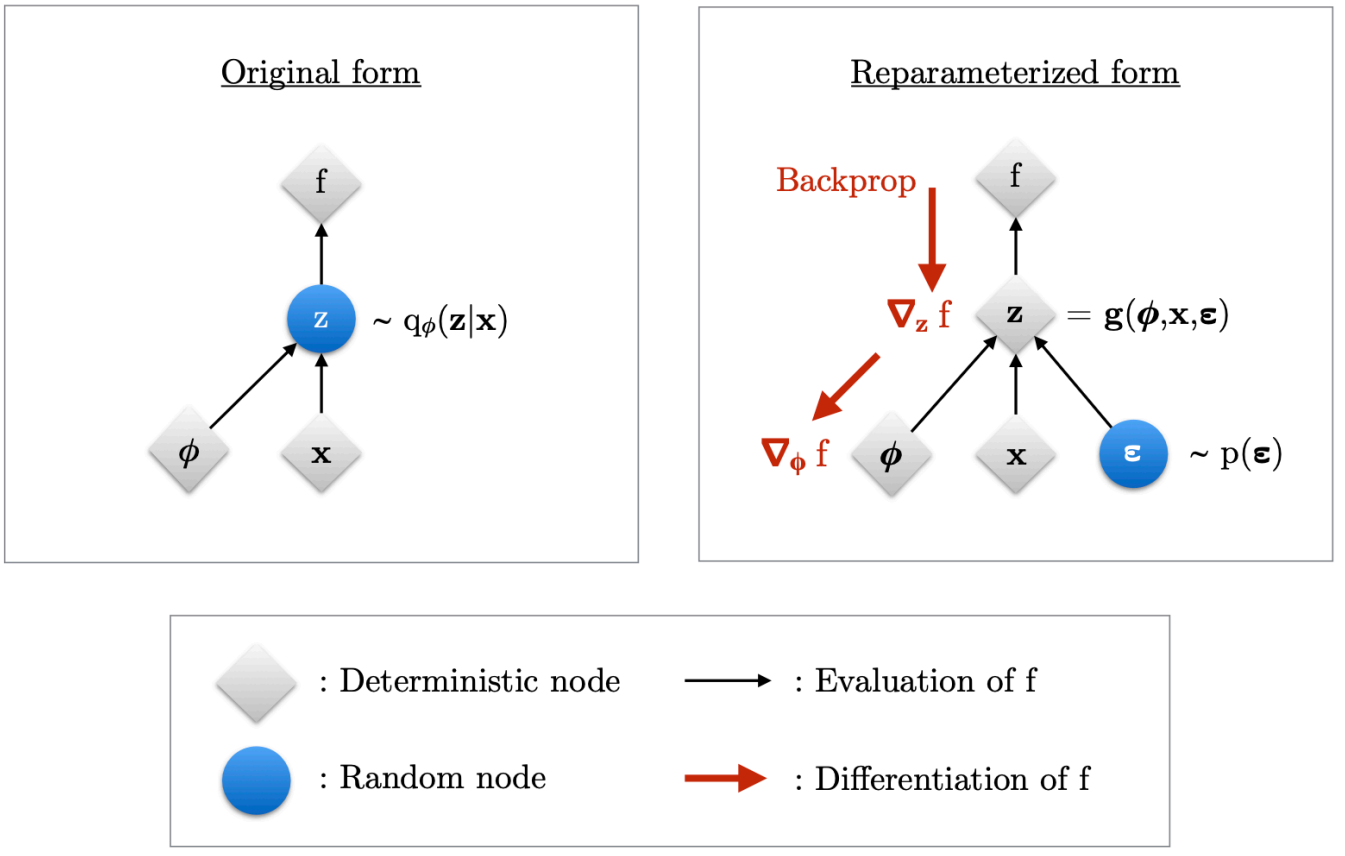
- High variance $\Rightarrow$ slow convergence
- Requires variance reduction techniques

When possible, we prefer the reparameterization trick...

Solution 2: Reparameterization Trick

Key idea: Express random variable as deterministic transformation

$$z \sim q_\psi(z) \quad \Rightarrow \quad z = g(\psi, \epsilon), \quad \epsilon \sim p(\epsilon)$$



Solution 2: Reparameterization Trick

Key idea: Express random variable as deterministic transformation

$$z \sim q_\psi(z) \quad \Rightarrow \quad z = g(\psi, \epsilon), \quad \epsilon \sim p(\epsilon)$$

Now gradient flows through g:

$$\nabla_\psi \mathbb{E}_{q_\psi}[f(z)] = \mathbb{E}_{p(\epsilon)}[\nabla_\psi f(g(\psi, \epsilon))]$$

Benefits:

- Low variance gradients
- Straightforward to implement
- Core technique for modern VI (VAEs, etc.)

Reparameterization: Gaussian Example

Gaussian distribution: $z \sim \mathcal{N}(\mu_\psi, \sigma^2_\psi)$

Reparameterization:

$$z = \mu_\psi + \sigma_\psi \cdot \epsilon, \qquad \epsilon \sim \mathcal{N}(0, 1)$$

Now z is a *differentiable function* of $\psi = (\mu_\psi, \sigma_\psi)$

Gradient computation:

- Sample $\epsilon \sim \mathcal{N}(0, 1)$
- Compute $z = \mu_\psi + \sigma_\psi \epsilon$
- Backpropagate through μ_ψ, σ_ψ

Reparameterization in Code i

Standard sampling (not differentiable):

```
z = np.random.normal(mu, sigma)  # Can't backprop!
```

Reparameterized sampling (differentiable):

```
eps = np.random.randn(*mu.shape)  # Sample noise
z = mu + sigma * eps # Deterministic transform
```

[Code S4 + S5]

Reparameterization in Code ii

Computing ELBO gradient:

```
def elbo_gradient(mu, sigma, log_joint):
    eps = np.random.randn(n_samples)
    z = mu + sigma * eps  # Reparameterization
    log_q = -0.5*np.log(2*np.pi*sigma**2) \
            - (z-mu)**2/(2*sigma**2)
    return np.mean(log_joint(z) - log_q)
    # Autodiff handles gradient through mu, sigma
```

Common Choice: Mean-Field Approximation

Mean-field assumption: Factorize over dimensions

$$q(z) = \prod_{i=1}^{D} q_i(z_i)$$

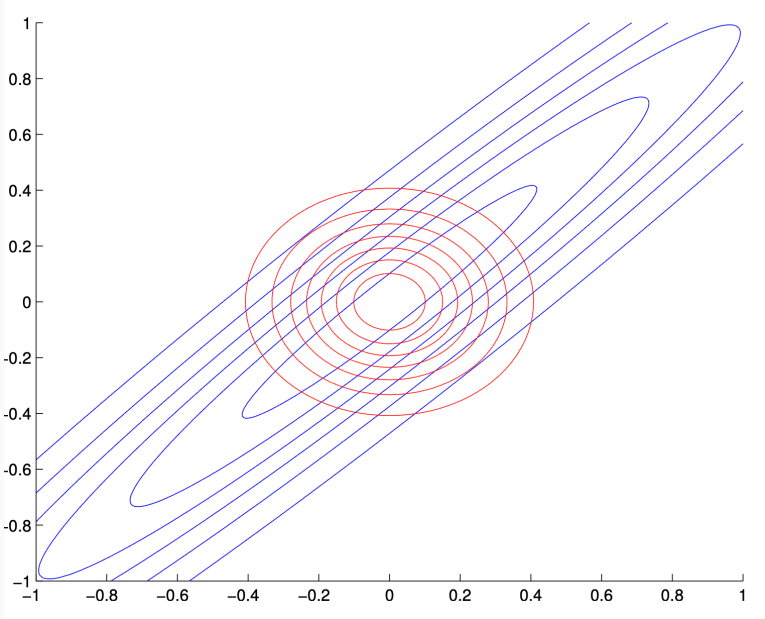
Why use mean-field?

- Dramatically reduces parameters: $O(D)$ vs $O(D^2)$
- Simplifies optimization
- Often sufficient for many applications

Example: Mean-field Gaussian

$$q(z) = \prod_{i=1}^{D} \mathcal{N}(z_i \mid \mu_i, \sigma_i^2)$$

Mean-Field Limitation: Ignoring Correlations



Issue: True posterior may have correlations

- Mean-field forces diagonal covariance
- Approximation is axis-aligned
- May poorly capture tilted/correlated structure

[Code S6]

Mean-Field Failure Mode: Variance Underestimation

Recall: We minimize $\mathrm{KL}(q\|p)$

Consequence:

- $\mathrm{KL}(q\|p)$ heavily penalizes q having mass where p is small
- Encourages q to be *narrower* than p
- Results in overconfident, "peaked" approximations

Example: Correlated Gaussian

$$p(z) = \mathcal{N}\left(0, \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}\right) \quad \Rightarrow \quad q(z) = \mathcal{N}(z_1 \mid 0, \sigma_1^2) \cdot \mathcal{N}(z_2 \mid 0, \sigma_2^2)$$

Mean-field q cannot capture the correlation and underestimates variance

Mean-Field Example: Code

True posterior: Correlated Gaussian

```
Sigma_true = np.array([[1.0, 0.8], [0.8, 1.0]])
```

Mean-field approximation: Diagonal covariance only

```
# MFVI restricts to diagonal
Sigma_mf = np.diag([sigma1**2, sigma2**2])
# Cannot represent off-diagonal correlation!
```

Despite this limitation, mean-field VI remains widely used due to its simplicity and efficiency

Contents

Scaling Challenge: Large Datasets

Problem: ELBO involves full dataset

$$\mathcal{L}(q) = \sum_{n=1}^{N} \mathbb{E}_q[\log p(x_n \mid z)] - \mathrm{KL}(q \| p(z))$$

Computing gradient requires:

- Iterating over all N data points
- Infeasible for large N (millions/billions of examples)

Idea: Can we use minibatches instead?

- Subsample data points
- Get unbiased gradient estimates
- Scale to massive datasets

Stochastic Variational Inference (SVI)

Key insight: Use stochastic optimization

Unbiased gradient estimator:

1. Sample minibatch $\mathcal{B} \subset \{1, \ldots, N\}$ with $|\mathcal{B}| = M$
2. Compute gradient on minibatch:

$$\nabla_\psi \mathcal{L}_\mathcal{B} = \frac{N}{M} \sum_{n \in \mathcal{B}} \nabla_\psi \mathbb{E}_{q_\psi}[\log p(x_n \mid z)] - \nabla_\psi \mathsf{KL}(q_\psi \| p)$$

3. Update: $\psi \leftarrow \psi + \rho \nabla_\psi \mathcal{L}_\mathcal{B}$

Combines:

- Reparameterization trick (low variance)
- Stochastic gradient descent (scalability)

SVI: Practical Algorithm

```
# Initialize variational parameters
lambda = initialize()

for iteration in range(max_iters):
    # Sample minibatch
    batch = sample_minibatch(data, batch_size)

    # Estimate gradient using reparameterization
    grad = estimate_gradient(lambda, batch)

    # SGD/Adam update
    lambda = optimizer.step(lambda, grad)
```

Key advantage: Each iteration is $O(M)$, not $O(N)$

[Code S7]

Contents

Free-Form VI: A Different Philosophy

Recall fixed-form VI:

- Choose $q_\psi(z)$ family (e.g., Gaussian)
- Optimize ψ via gradients

Free-form VI asks:

What if we don't choose the form of q upfront?

Approach:

- Assume factorization: $q(z) = \prod_i q_i(z_i)$
- Use *calculus of variations* to derive optimal q_i^*
- Results in iterative coordinate updates

This leads to Coordinate Ascent Variational Inference (CAVI)

Coordinate Ascent VI (CAVI): The Update Rule

Optimal factor q_i^* has closed form:

$$\log q_i^*(z_i) = \mathbb{E}_{q_{\neg i}}[\log p(x, z)] + C$$

where $q_{\neg i} = \prod_{j \neq i} q_j(z_j)$

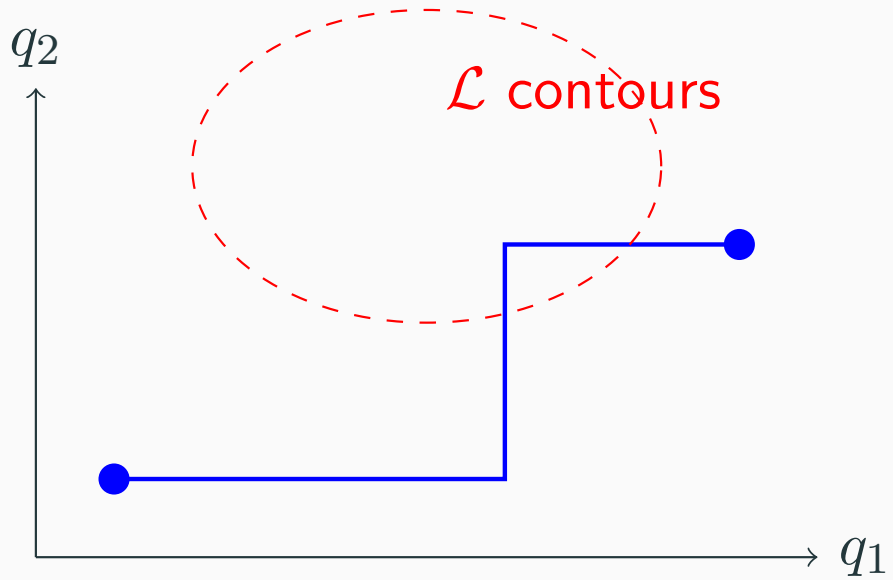
Intuition:

- Hold all other factors fixed
- Update q_i to its "best response"
- Take expectation over other latent variables
- Alternate updates until convergence

Requirement: Model must be conjugate (exponential family)

CAVI Intuition

Think of it as coordinate ascent on the ELBO:



- Each update: maximize ELBO w.r.t. one factor
- ELBO increases monotonically
- Guaranteed to reach local optimum

CAVI Example: Gaussian Mixture Model

Latent variables:

- z_n: cluster assignment for data point n
- μ_k: cluster means
- π: mixing proportions

CAVI alternates updates:

1. Update responsibilities: $q(z_n) \propto \exp\{\mathbb{E}[\log p(x_n, z_n \mid \mu, \pi)]\}$
2. Update cluster means: $q(\mu_k) \propto \exp\{\mathbb{E}[\log p(\mathbf{x}, z, \mu_k)]\}$
3. Update mixing weights: $q(\pi) \propto \exp\{\mathbb{E}[\log p(\mathbf{x}, z, \pi)]\}$

Conjugacy $\Rightarrow$ All updates are closed-form!

CAVI: General Algorithm

```
# Initialize all factors
q = initialize_factors()

while not converged:
    for i in latent_variables:
        # Update factor i given all others
        q[i] = compute_optimal_factor(
            q_except_i=q[:i] + q[i+1:],
            joint_log_prob=log_p
        )

    # Check ELBO convergence
    if elbo_change < tolerance:
        break
```

[Code S8]

CAVI: Convergence Behavior

Guarantees:

- ELBO increases monotonically: $\mathcal{L}^{(t+1)} \geq \mathcal{L}^{(t)}$
- Converges to a local maximum
- Deterministic updates (reproducible)

Practical considerations:

- Initialization matters (multiple random starts)
- May converge to poor local optima
- Slower than gradient-based VI for high-dimensional problems

Limitation: Requires full dataset pass per update
(Can we make CAVI scalable too? Yes! Stochastic variants exist)

Contents

Method Comparison

Method	Family Choice	Scalability	Needs Conjugacy
Fixed-Form	Parametric q_ψ	With SVI	No
CAVI	Derived from $\mathcal{L}$	No*	Yes
SVI	Parametric q_ψ	Yes	No

*Stochastic CAVI variants exist

Method Comparison

Practical guidance:

- **Large scale + flexible model:** SVI with reparameterization
- **Conjugate exponential family:** CAVI
- **Complex posterior structure:** Normalizing flows, neural VI

The landscape:

- Fixed-form VI: flexible, scalable, works anywhere
- CAVI: elegant closed-form when conjugate
- SVI: best of both worlds for large-scale problems

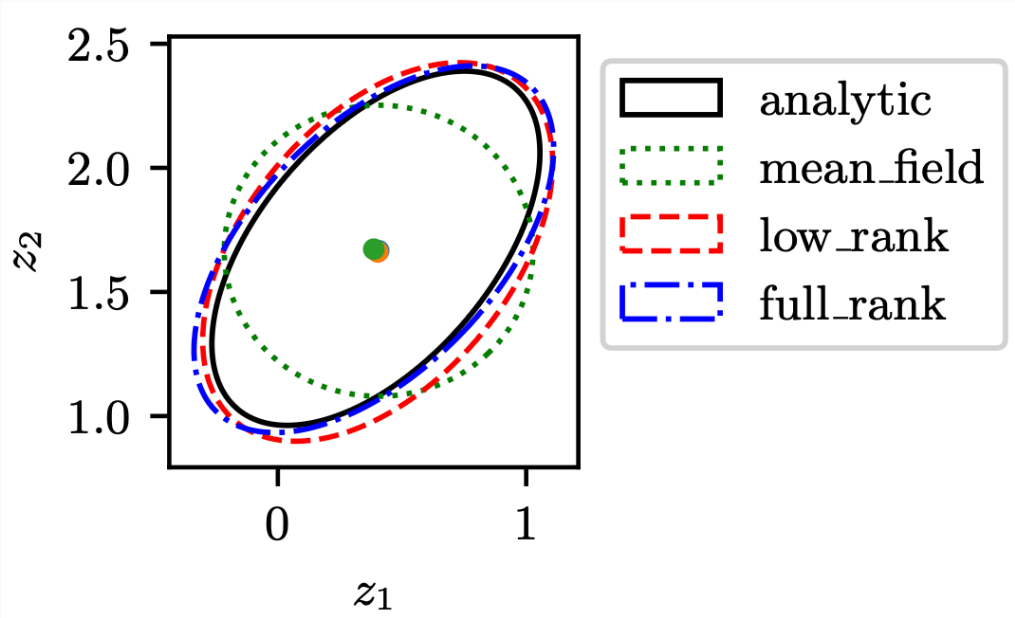
Contents

Limitations of MFVI

- Underestimates posterior variance.
- Cannot capture correlations.
- KL(q‖p) is mode-seeking.

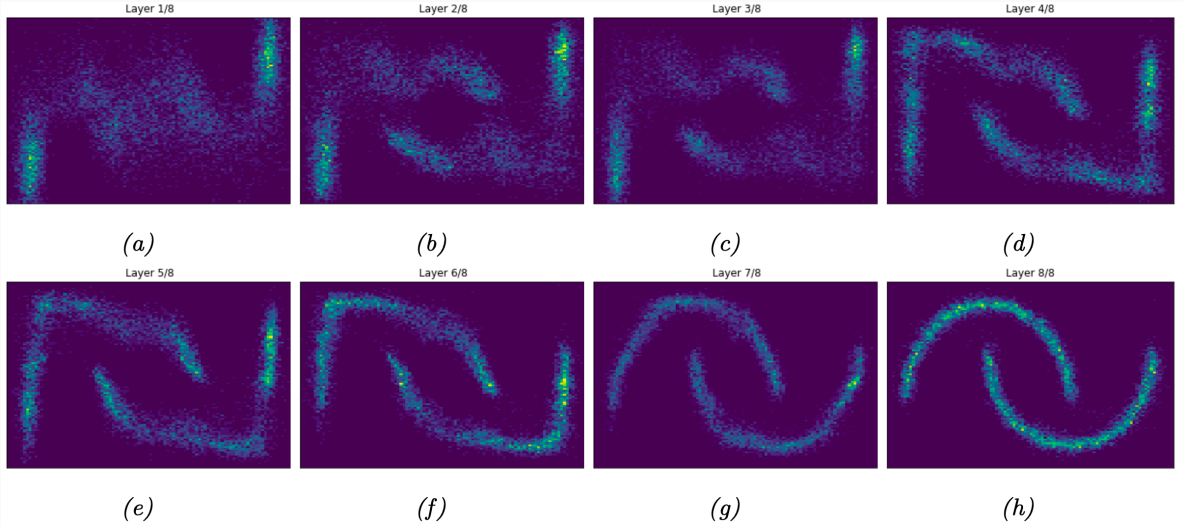
Structured Variational Families

- Add correlations.
- Tree-structured VI.
- Matrix-variate Gaussians.



Normalizing Flows

- Transform simple q(z) to complex distribution.
- Invertible mapping with tractable Jacobian.



[Code S10 + S11]

Extra: Score-Matching VI

Key idea: Instead of matching densities (ELBO), match *score functions*:

$$\nabla_z \log q_\lambda(z) \approx \nabla_z \log p(z|x).$$

Score-Matching VI (GSM-VI):

- Iteratively adjusts q_λ to match posterior scores at sampled points.
- Closed-form updates when q_λ is Gaussian.

Pros:

- Black-box (only needs differentiable joint).
- Often 10–100$\times$ fewer gradients than ELBO methods.

Contents

Key Takeaways

Core concepts:

1. VI converts inference into optimization via ELBO
2. Two paradigms: fixed-form vs free-form
3. Reparameterization enables low-variance gradients
4. Mean-field simplifies but loses correlations
5. Stochasticity enables scalability

Extra refs

Blei, D. et al. (2017). *Variational inference: A review for statisticians*. JASA.

Knoblauch, J. et al. (2022). *An Optimization-centric View on Bayes' Rule*. JMLR.

Modi, C. et al. (2023). *Variational inference with Gaussian score matching*. NeurIPS.

Questions?

Next Session

Monte Carlo & Hamiltonian Methods (Ch. 12)

Dec 3, 2025

Miguel Santos