The Reparameterization Trick and Auto-Encoding Variational Bayes

N-20251011

§1.1 The gradient problem

The ELBO for one data point is

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x \mid z)] - D_{\mathrm{KL}}(q_\phi(z \mid x) \| p(z)).$$

For the diagonal Gaussian encoder and standard Gaussian prior, the KL term has a closed form. The difficult part is the expectation

$$\mathbb{E}_{z \sim q_\phi(z|x)}[\log p_\theta(x \mid z)].$$

We need gradients with respect to both θ and ϕ:

$$\nabla_{\theta,\phi} \mathbb{E}_{z \sim q_\phi(z|x)}[\log p_\theta(x \mid z)].$$

The subtlety is that ϕ controls the distribution from which z is sampled. The sample z is random, but its randomness depends on the encoder parameters.

The reparameterization trick is the mathematical device that turns this stochastic node into a differentiable transformation of parameter-free noise.

§1.2 Score-function estimator and pathwise estimator

There is a general identity for differentiating expectations:

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

This is often called a score-function estimator. It is very general because it does not require z to be a differentiable function of ϕ. But it often has high variance.

The reparameterization trick uses a more geometric idea. Instead of sampling

$$z \sim q_\phi(z \mid x),$$

write

$$z = g_\phi(\epsilon, x), \qquad \epsilon \sim p(\epsilon),$$

where the noise distribution $p(\epsilon)$ does not depend on ϕ.

Then

$$\mathbb{E}_{z \sim q_\phi(z|x)}[f(z)] = \mathbb{E}_{\epsilon \sim p(\epsilon)}[f(g_\phi(\epsilon, x))].$$

Now the parameters appear inside an ordinary differentiable function. Gradients can pass through g_ϕ.

§1.3 Gaussian reparameterization

For the standard VAE encoder,

$$q_\phi(z \mid x) = \mathcal{N}(z; \mu_\phi(x), \mathrm{diag}(\sigma_\phi^2(x))).$$

A sample from this distribution can be written as

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I).$$

Here $\odot$ denotes coordinatewise multiplication.

Thus

$$\begin{aligned}&\mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x \mid z)] \\ &\quad = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}\left[\log p_\theta(x \mid \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon)\right].\end{aligned}$$

The distribution of ϵ is fixed. The dependence on ϕ is now inside μ_ϕ and σ_ϕ, so ordinary backpropagation can be used.

The trick is not a heuristic. It is a change of variables in the sampling procedure.

§1.4 The stochastic ELBO estimator

With L samples $\epsilon^{(1)}, \ldots, \epsilon^{(L)} \sim \mathcal{N}(0, I)$, define

$$z^{(\ell)} = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon^{(\ell)}.$$

Then the Monte Carlo estimator of the ELBO is

$$\widehat{\mathcal{L}}(\theta, \phi; x) = \frac{1}{L}\sum_{\ell=1}^{L} \log p_\theta(x \mid z^{(\ell)}) - D_{\mathrm{KL}}(q_\phi(z \mid x) \| p(z)).$$

For minibatches, the dataset objective is approximated by averaging this quantity over the examples in the minibatch.

The original AEVB method is therefore a stochastic-gradient method for maximizing a variational lower bound. The randomness is not removed; it is organized so that differentiation is possible.

§1.5 Why implementations output log variance

Mathematically, the encoder needs $\sigma_\phi(x) > 0$. In practice, one often outputs

$$\log \sigma_\phi^2(x)$$

instead of $\sigma_\phi(x)$ directly. If

$$\ell_\phi(x) = \log \sigma_\phi^2(x),$$

then

$$\sigma_\phi(x) = \exp\left(\frac{1}{2}\ell_\phi(x)\right).$$

This automatically keeps the standard deviation positive.

The Gaussian KL then becomes

$$D_{\mathrm{KL}}(q \| p) = \frac{1}{2}\sum_{j=1}^{d}\left(\mu_j^2 + \exp(\ell_j) - 1 - \ell_j\right).$$

This is the same formula as before, written in terms of log variance.

§1.6 Training as ELBO maximization

The mathematical training loop is:

encoder:	$x \mapsto \mu_\phi(x),\ \log \sigma^2_\phi(x),$
noise:	$\epsilon \sim \mathcal{N}(0, I),$
latent sample:	$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon,$
decoder:	$z \mapsto$ parameters of $p_\theta(x \mid z),$
objective:	maximize $\widehat{\mathcal{L}}(\theta, \phi; x).$

If one minimizes a loss instead, the loss is the negative ELBO:

$$-\widehat{\mathcal{L}} = -\frac{1}{L}\sum_{\ell=1}^{L} \log p_\theta(x \mid z^{(\ell)}) + D_{\mathrm{KL}}(q_\phi(z \mid x)\|p(z)).$$

The first term depends on the decoder likelihood. The second term is analytic for the standard Gaussian encoder.

§1.7 Decoder likelihoods and reconstruction losses

For binary data, a Bernoulli decoder gives

$$p_\theta(x \mid z) = \prod_i \pi_i(z)^{x_i}(1 - \pi_i(z))^{1-x_i}.$$

The negative log-likelihood is

$$-\log p_\theta(x \mid z) = -\sum_i \left[x_i \log \pi_i(z) + (1 - x_i)\log(1 - \pi_i(z))\right].$$

This is binary cross entropy.

For real-valued data, a Gaussian decoder with fixed variance gives

$$p_\theta(x \mid z) = \mathcal{N}(x; \mu_\theta(z), \sigma^2 I).$$

Then

$$-\log p_\theta(x \mid z) = \frac{1}{2\sigma^2}\|x - \mu_\theta(z)\|^2 + \text{constant}.$$

This is squared error up to scaling and constants.

Thus the word "reconstruction loss" hides a probabilistic choice. Different likelihood models produce different reconstruction terms.

§1.8 Generation after training

After learning θ, generation follows the model direction, not the inference direction:

$$z \sim p(z), \qquad x \sim p_\theta(x \mid z).$$

If the decoder likelihood is Gaussian, one may sample x from that Gaussian or use its mean as a representative output. If the decoder likelihood is Bernoulli, one samples binary coordinates or uses probabilities for visualization.

Latent interpolation uses points such as

$$z_t = (1 - t)z_0 + tz_1, \qquad 0 \le t \le 1.$$

This often produces visually smooth transitions, but it is not a theorem that every linear path in latent space is semantically meaningful. It works when the learned latent geometry is reasonably aligned with the prior and decoder.

§1.9 Empirical marginal likelihood curves

The original experiments compared AEVB with older methods by plotting marginal log-likelihood estimates. The figure is included not as the main point of the theory, but as a reminder that the ELBO machinery was designed to make likelihood-based generative modeling trainable.

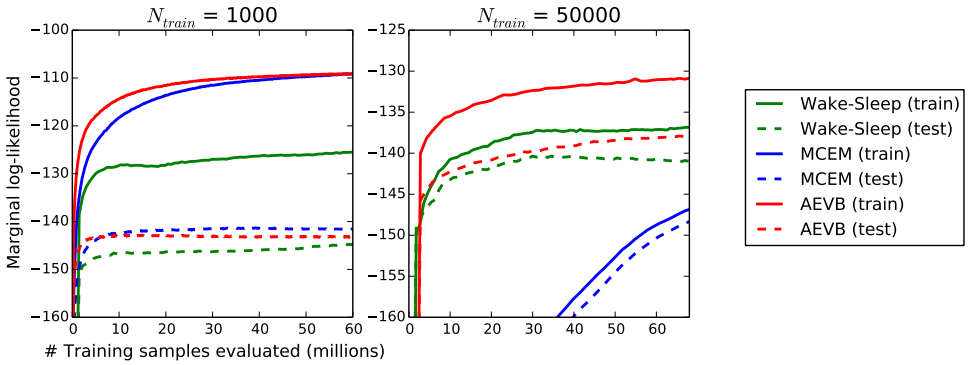

Figure 1.1: Marginal log-likelihood curves in the AEVB experiments.

The mathematical object behind these curves is still the same:

$$\log p_\theta(x) = \mathcal{L}(\theta, \phi; x) + D_{\mathrm{KL}}(q_\phi(z \mid x) \| p_\theta(z \mid x)).$$

Training improves a tractable stochastic estimate of $\mathcal{L}$, and the hope is that this also improves the marginal likelihood.

§1.10 Limitations and extensions

Several limitations are already visible from the mathematics.

First, the posterior family may be too simple. A diagonal Gaussian cannot capture all posterior shapes. Normalizing flows enrich $q_\phi(z \mid x)$ by transforming a simple base distribution through invertible maps.

Second, the prior may be too simple. The standard normal prior is convenient, but the aggregated posterior may have more complicated geometry.

Third, posterior collapse may occur. If the decoder is too expressive, the model may learn

$$q_\phi(z \mid x) \approx p(z),$$

so that z carries little information about x. Then the KL term is small, but the latent representation is weak.

Fourth, the usual ELBO uses one particular weighting between reconstruction and KL. Variants such as β-VAE modify the objective:

$$\mathbb{E}_q[\log p_\theta(x \mid z)] - \beta D_{\mathrm{KL}}(q_\phi(z \mid x) \| p(z)).$$

This changes the rate–distortion tradeoff and can encourage more disentangled latent coordinates, though not automatically.

§1.11 How the pieces fit

The VAE becomes trainable because the stochastic latent sample is rewritten as

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I).$$

This makes the expectation differentiable with respect to the encoder parameters:

$$\mathbb{E}_{z \sim q_\phi(z|x)}[f(z)] = \mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)}[f(\mu_\phi(x) + \sigma_\phi(x) \odot \epsilon)].$$

The full mathematical chain is now visible:

$$\begin{gathered}\text{latent variable model } p(z)p_\theta(x \mid z) \\ \Downarrow \\ \text{intractable posterior } p_\theta(z \mid x) \\ \Downarrow \\ \text{variational approximation } q_\phi(z \mid x) \\ \Downarrow \\ \text{ELBO lower bound} \\ \Downarrow \\ \text{reparameterized stochastic gradients.}\end{gathered}$$

This is why the VAE is more than an autoencoder with noise. It is a method for turning approximate Bayesian inference in a latent variable model into differentiable optimization.