Lecture 2a: Bayesian inference by gradient descent

In the previous chapter, we discussed continuous Bayesian inference in one dimension. We saw that in most realistic cases the inference can be approximated in a tractable way using quantization and that in some special cases we can obtain closed form formulas for the Bayesian updates. In this chapter, we will learn how to approximate a univariate Bayesian posterior by gradient descent. While this is usually not the best approach in low-dimensional cases, it will lay the foundation of the general multivariate variational algorithms that awe are going to discuss in the next chapter and in the rest of the book.
As an example, consider a simple Gaussian inference problem defined by the following distributions

$$p(D \mid x) = \prod_{k=1}^{K} \mathcal{N}(y_k; x, 1) \ , \tag{1}$$

$$p(x) = \mathcal{N}(x; 0, 1) \ , \tag{2}$$

where D is a dataset of comprised of K univariate measurements of a scalar quantity y. Of course, in this case we know how to compute the posterior $p(x \mid D)$ is closed form. However, let us pretend to not know the exact posterior and try to obtain the solution by gradient descent.
The first step of any gradient descent algorithm is to define a parameterized model of the solution. In this case, we can parameterize the solution as a Gaussian distribution parameterized by mean μ and variance ν^2:

$$q(x; \mu, \nu) = \mathcal{N}(x; \mu, \nu^2) \ . \tag{3}$$

in the rest of the book, we will refer to this parameterized model as *variational posterior*, *variational approximation* or *variational sampler*. The hope is that we can find the true posterior by tuning the parameters μ and ν, or at least that we can find a good approximation of this form. The second step in any gradient descent algorithm is to define an appropriate loss function $\mathcal{L}(\mu, \nu)$. Ideally, the loss should be equal to zero when the model perfectly matches the exact solution and should be greater than zero otherwise. As we learned in the last chapter, these properties are met by the KL divergence between the exact posterior and the parameterized approximation:

$$\mathcal{L}(\mu, \nu) = -\int_{-\infty}^{\infty} \log \frac{p(x \mid D)}{q(x; \mu, \nu)} q(x; \mu, \nu) \mathrm{d}x \ . \tag{4}$$

At first sight, this loss function does not seem to be very useful since it can only be evaluated if we know the exact posterior $p(x \mid D)$. Needless to say, this would defy the purpose as if the exact posterior is already available no approximation is needed!
However, here is still hope. In order to implement a gradient descent algorithm, we only need to evaluate the gradient $\nabla_{\mu,\nu}\mathcal{L}(\mu,\nu)$ of the loss function while we do not need to evaluate the loss itself. Remember that the only intractable term in the posterior is the normalization integral $p(D)$ and that the logarithm of the posterior is given by

$$\log p(x \mid D) = \log p(D \mid x)p(x) - \log p(D) \ , \tag{5}$$

where $p(D \mid x)p(x)$ is just the product between likelihood and prior. Using this formula, we can evaluate the gradient of the KL divergence:

$$\begin{aligned}
\nabla_{\mu,\nu}\mathcal{L}(\mu,\nu) &= -\nabla_{\mu,\nu}\int_{-\infty}^{\infty}\left(\log\frac{p(D \mid x)p(x)}{q(x;\mu,\nu)} - \log p(D)\right)q(x;\mu,\nu)\mathrm{d}x \ , \qquad (6)\\
&= -\nabla_{\mu,\nu}\int_{-\infty}^{\infty}\log\frac{p(D \mid x)p(x)}{q(x;\mu,\nu)}q(x;\mu,\nu)\mathrm{d}x + \int_{-\infty}^{\infty}\log p(D)q(x;\mu,\nu)\mathrm{d}x \ ,\\
&= -\nabla_{\mu,\nu}\int_{-\infty}^{\infty}\log\frac{p(D \mid x)p(x)}{q(x;\mu,\nu)}q(x;\mu,\nu)\mathrm{d}x + \log p(D)\nabla_{\mu,\nu}\int_{-\infty}^{\infty}q(x;\mu,\nu)\mathrm{d}x \ ,\\
&= -\nabla_{\mu,\nu}\int_{-\infty}^{\infty}\log\frac{p(D \mid x)p(x)}{q(x;\mu,\nu)}q(x;\mu,\nu)\mathrm{d}x + \log p(D)\nabla_{\mu,\nu}1 \ ,\\
&= -\nabla_{\mu,\nu}\int_{-\infty}^{\infty}\log\frac{p(D \mid x)p(x)}{q(x;\mu,\nu)}q(x;\mu,\nu)\mathrm{d}x \ . \qquad (7)
\end{aligned}$$

This is very promising! The intractable normalization integral does not play any role in the gradient since it simply add a constant shift independent of the variational parameters μ and ν. However, it is not clear if this expression is tractable since it still involves the solution of an integral. Is this integral easier than the original normalization integral $p(D) = \int p(D \mid x)p(x)\mathrm{d}x$? If not, we would have only achieved to convert the solution of an integral into a gradient descent algorithm that requires us to solve a brand new challenging integrals for each update step!

1 Tractable and intractable integrals

As we discussed in the previous chapter, there are two sources of intractability in univariate integrals: 1) The function to integrate could very quickly and 2) the function to integrate has all non-vanishing values in a small region with unknown location. For example, in the Gaussian model given in Eq. 4, the normalization

integral has the following form

$$p(D) = \int_{-\infty}^{\infty} f(x)\mathrm{d}x \ , \tag{8}$$
$$= c_p \int_{-\infty}^{\infty} e^{-(x-\mu_p)^2/2\sigma_p^2}\mathrm{d}x \ ,$$

where c_p, μ_p and σ_p^2 are constants. In this case, we know that the function is Gaussian and we can explicitly evaluate all the constants. However, consider the scenario where we can evaluate $f(x)$ for any value of x but where we do not have knowledge about the location of its peak. Since $f(x)$ is a Gaussian function, only a negligible amount of density lays outside the interval $(\mu_p - 5\sigma_p, \mu_p + 5\sigma_p)$. Now imagine that μ_p could be any value between -10^20 and 10^20 and that σ_p is equal to one. Finding the small interval in this enormous range just by evaluating the function at random points is a hopeless task. Similarly, if we try to solve the integral by quantization we will need to use an impossible number of grid points in order to not miss the only important region.
Let us now consider the integral in Eq. 7 needed to implement the gradient descent algorithm:

$$\int_{-\infty}^{\infty} \log \frac{f(x)}{q(x;\mu,\nu)} q(x;\mu,\nu)\mathrm{d}x \tag{9}$$
$$= \int_{-\infty}^{\infty} \left(-\frac{1}{2\sigma_p^2}(x-\mu_p)^2 + \frac{1}{2\sigma^2}(x-\mu)^2 - \log c_p - \log 2\pi\sigma^2 \right) q(x;\mu,\nu)\mathrm{d}x \ .$$

Again, we pretend to not know where μ_p is. However, we can safely assume to know μ and σ since they are the parameters we are updating through gradient descent. The function to integrate is now the product between a quadratic function

$$\log \frac{f(x)}{q(x;\mu,\nu)} = \left(-\frac{1}{2\sigma_p^2}(x-\mu_p)^2 + \frac{1}{2\sigma^2}(x-\mu)^2 - \frac{1}{2}\log c_p - \log 2\pi\sigma^2 \right) \tag{10}$$

and a Gaussian known density $q(x;\mu,\nu)$. From your calculus classes, you probably remember that when an exponential (such as the e^{-x^2} in the Gaussian density) wants to go to zero there is nothing that a polynomial can do to stop it, let alone a humble square function. This implies that the function in this integral is non-vanishing wherever the variational distribution $q(x;\mu,\nu)$ is non-vanishing, namely in an interval such as $(\mu - 5\sigma, \mu + 5\sigma)$. It is now clear that approximating the integral is easy since we can discretize this interval into a relative small number of bins and neglect everything outside as its contribution is negligible. While we showed this in the context of Gaussian inference, this phenomenon is very general since the logarithm will almost always "tame" the extreme behavior of $f(x)$ by transforming a super-exponential vanishing into a gentler sub-exponential descent to $-\infty$ which is usually counteracted by the exponential vanishing of the variational distribution. This is visualized in Fig. 1.

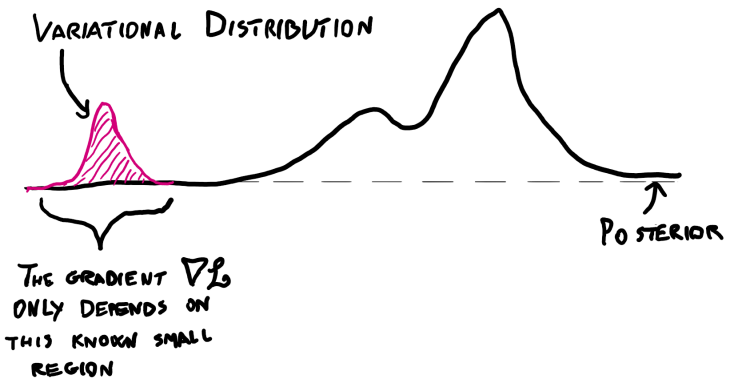

Figure 1: The locality of variational gradient descent.

2 Variational gradient descent

It should be now clear that the integral in our gradient in Eq. 7 is in general much more tractable than the normalization integral. This justifies the use of gradient descent on a KL loss function for approximate Bayesian inference. From now on, I will refer to this approach as variational gradient descent. There is a very strict connection between this Bayesian gradient descent algorithm and the regular gradient descent you probably encountered in introductory optimization and machine learning books. To see this, consider a variational Gaussian approximation $q_\sigma(x \mid \mu)$ with learnable mean and fixed standard deviation σ. Let us initialize the algorithm at an arbitrary initial parameter μ_0. The update step is given by the usual rule

$$\mu_{n+1} = \mu_n - \eta \frac{\partial}{\partial \mu} \mathcal{L}_\sigma(\mu) \ , \tag{11}$$

where η is a small learning rate. Note that, while we are not training the scale parameter σ, its value still affect the form of the loss function. Up to irrelevant terms that do not depend on μ, the loss $\mathcal{L}(\mu)$ is given by the integral

$$\begin{aligned} \mathcal{L}_\sigma(\mu) &= -\int_{-\infty}^{\infty} \log \frac{f(x)}{q(x;\mu,\nu)} q_\sigma(x;\mu) \mathrm{d}x \\ &= \int_{-\infty}^{\infty} \left(\log f(x) + \log q_\sigma(x;\mu)\right) q_\sigma(x;\mu) \mathrm{d}x \ , \\ &= \int_{-\infty}^{\infty} \log f(x) q_\sigma(x;\mu) \mathrm{d}x + \mathcal{DH}\left[q_\sigma(x;\mu)\right] \end{aligned} \tag{12}$$

where

$$\mathcal{DH}\left[q_\sigma(x;\mu)\right] = \int_{-\infty}^{\infty} q_\sigma(x \mid \mu) \log q_\sigma(x \mid \mu) \mathrm{d}x \tag{13}$$

is the differential entropy of the variational distribution. Since the distribution is Gaussian, this entropy is equal to $(1/2)\log \pi e \sigma^2$ and, since this term does not depend on μ, the gradient becomes:

$$\frac{\partial}{\partial \mu}\mathcal{L}_\sigma(\mu) = \frac{\partial}{\partial \mu}\int_{-\infty}^{\infty} \log f(x) q_\sigma(x;\mu)\mathrm{d}x \ . \tag{14}$$

As we explained above, the non-vanishing region of $q_\sigma(x;\mu)$ determines the range of x in which the values $f(x)$ contribute to the integral. At the limit $\sigma \to 0$, this relevant region shrinks around μ as $q_\sigma(x;\mu)$ converges to the deterministic distribution $\delta(x-\mu)$, which has all its probability density on the single value μ. Therefore

$$\lim_{\sigma\to 0}\frac{\partial}{\partial \mu}\mathcal{L}_\sigma(\mu) = \lim_{\sigma\to 0}\frac{\partial}{\partial \mu}\int_{-\infty}^{\infty} \log f(x) q_\sigma(x;\mu)\mathrm{d}x = \frac{\partial \log f(\mu)}{\partial \mu} \ . \tag{15}$$

To summarize, at the limit of zero variational variance the vocational gradient of the mean parameter is identical to the regular gradient of the joint distribution $f(x) = p(D \mid x)p(x)$. This establishes a close connection between variational inference and the maximum-a-posteriori (MAP) algorithm. As you will see in later chapters, this close connection between variational inference and gradient descent will allow us to use all the arsenal of deep learning techniques for solving and defining Bayesian problems.

2.1 Stochastic gradients and the reparameterization trick

At this point, I hope I managed to convince you that variational inference by gradient descent is a promising approach. However, I still did not define a feasible algorithm since the integral in Eq. 9 cannot in general be evaluated analytically. In this section, I will show you how to write the gradient the the variational loss as a expectation with respect to a fixed distribution, which can in turn be approximated using a finite number of samples. In the Gaussian problem defined by Eq. 4, the variational loss is

$$\mathcal{L}(\mu,\sigma) = \underbrace{\int_{-\infty}^{\infty} \frac{1}{2}\left(\sum_{k=1}^{K}(x-y_k)^2 + x^2\right)\frac{e^{-(x-\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}}\mathrm{d}x}_{\frac{1}{2}\mathbb{E}_{x\sim q(x;\mu,\sigma)}\left[\sum_{k=1}^{K}(x-y_k)^2+x^2\right]} - \underbrace{\frac{1}{2}\log \pi e \sigma^2}_{\text{Diff. Entropy}} \ . \tag{16}$$

Note that the integral is the expected value of the function $\left(\sum_{k=1}^{K}(x-y_k)^2 + x^2\right)$ with respect to the variational distribution. This realization offers a strategy to approximate the integral. We can sample a N values of x from q and then replace the exact expectation with a finite average:

$$\mathbb{E}_{x\sim q(x;\mu,\sigma)}\left[\sum_{k=1}^{K}(x-y_k)^2 + x^2\right] \approx \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{k=1}^{K}(x_n-y_k)^2 + x_n^2\right) \ . \tag{17}$$

This is a so called *unbiased stochastic estimator* of the expectation integral. The term unbiased means that the estimator gives us the exact value of the integral if we average of infinitely many re-samplings of the vales of x. Furthermore, the variance of the estimator tends to zero as N tends to infinity. We can therefore try to approximate the exact gradient using this sampling approach. If we try this for the mean parameter, we get

$$\begin{aligned}
&\frac{\partial}{\partial \mu}\mathcal{L}(\mu,\sigma) \approx \qquad (18)\\
&= \frac{1}{2N}\sum_{n=1}^{N}\left(\sum_{k=1}^{K}\frac{\partial}{\partial\mu}(x_n - y_k)^2 + \frac{\partial}{\partial\mu}x_n^2\right) - \frac{\partial}{\partial\mu}\frac{1}{2N}\log\pi e\sigma^2 \ ,\\
&= \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{k=1}^{K}(x_n - y_k) + x_n\right)\frac{\partial x_n}{\partial\mu} \ .
\end{aligned}$$

This expression accounts for the fact that each sample x_n depends on μ since it has been sampled from the distribution $q(x;\mu,\sigma)$, which in turns depends on μ. How can we compute the derivative $\frac{\partial x_n}{\partial\mu}$? The trick is to express x_n as a function of the variational parameters μ and σ together with a random variable ϵ_n that does not depend on the parameters. Let us try therefore to express x_n as the translated and scaled version of a variable ϵ_n:

$$x_n = \mu + \sigma\epsilon_n \ . \qquad (19)$$

Using this expression, we can write the derivative of the sample as:

$$\frac{\partial x_n}{\partial\mu} = 1 + \sigma\frac{\partial\epsilon_n}{\partial\mu} \ . \qquad (20)$$

This is not necessarily a progress since the distribution of ϵ_n could in turn depend on μ and we could therefore end up with another unknown derivative. However, it turns out that, if x_n is Gaussian, ϵ_n is a standard Gaussian, which does not depend on any parameter:

$$\epsilon_n = \frac{x_n - \mu}{\sigma} \sim \mathcal{N}(\epsilon_n;0,1) \ . \qquad (21)$$

Therefore, $\frac{\partial\epsilon_n}{\partial\mu}$ is equal to zero and consequentially:

$$\frac{\partial x_n}{\partial\mu} = 1 \ . \qquad (22)$$

This is the so called *reparameterization trick* . Note that this specific reparameterization formula will only work for a variational distribution obtained by translating and scaling a fixed base distribution. In the general case, the dependency of x_n on the parameters can be more complicated, which would require a more complex reparameterization formula. We can finally write down

the (reparameterization) gradient estimator in a fully explicit form:

$$\frac{\partial}{\partial \mu}\mathcal{L}(\mu,\sigma) \approx \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{k=1}^{K}(x_n - y_k) + x_n\right) \ . \tag{23}$$

Note that, in this simple Gaussian case, the estimator can be easily simplified. In fact

$$\sum_{n=1}^{N}\left(\sum_{k=1}^{K}(x_n - y_k) + x_n\right) = -\sum_{k=1}^{K} y_k + (K+1)\frac{1}{N}\sum_{n=1} x_n \ . \tag{24}$$

Note that $\frac{1}{N}\sum_{n=1} x_n$ is the empirical average of the samples x_n which clearly tends to μ for $N \to \infty$. Since we know that the estimator tends to the exact loss at this limit, we can conclude that

$$\frac{\partial}{\partial \mu}\mathcal{L}(\mu,\sigma) = \lim_{n\to\infty}\left(-\sum_{k=1}^{K} y_k + (K+1)\frac{1}{N}\sum_{n=1} x_n\right) = -K\bar{y} + (K+1)\mu \ , \tag{25}$$

where $\bar{y}$ is the average of the data. Hence, in this example the re-parameterization trick allowed us to compute the gradient exactly. However, this is not possible in most cases where we need to approximate the gradient by sampling a finite number of variational samples.
Note that in this simple case we do not even need to perform the gradient descent algorithm as we can optimize the loss in closed form by setting the derivative equal to zero:

$$\frac{\partial}{\partial \mu}\mathcal{L}(\mu,\sigma) = 0 \Longrightarrow \mu = \frac{K}{K+1}\bar{y} \ . \tag{26}$$

We can also evaluate the gradient for the standard deviation parameter by noticing that

$$\frac{\partial x_n}{\partial \sigma} = \frac{\partial}{\partial \sigma}\left(\mu + \sigma\epsilon_n\right) = \epsilon_n = (x_n - \mu)/\sigma \ , \tag{27}$$

which gives us

$$\begin{aligned} &\frac{\partial}{\partial \sigma}\mathcal{L}(\mu,\sigma) \approx \\ &\frac{1}{\sigma}\frac{1}{N}\sum_{n=1}^{N}\left(\sum_{k=1}^{K}\left((x_n - y_k)(x_n - \mu) + x_n(x_n - \mu)\right)\right) - \frac{1}{\sigma} \ . \end{aligned} \tag{28}$$

where the last term comes from the differential entropy. It is not too difficult to show (try it) that at the limit $N \to \infty$, the expression becomes:

$$\frac{\partial}{\partial \sigma}\mathcal{L}(\mu,\sigma) = (K+1)\sigma - 1/\sigma \ . \tag{29}$$

We can again set this derivative equal to zero and obtain the solution:

$$\sigma = \frac{1}{\sqrt{K+1}} \ . \tag{30}$$

We therefore got a full close form solution to a Gaussian inference problem using the variational approach. This of course is not surprising since we showed in the last chapter that this inference problem can indeed be solved in closed form. This is a confirmation of the tongue-in-cheek mathematical principle of *conservation of effort*: when the solution of a problem can be achieved using method X, it can as easily be obtained using any other appropriate method Y. Of course, most inference problems cannot be solved in closed form. In that case, the gradient has to be use in a gradient descent algorithm.

3 Stochastic variational gradient descent for Gaussian models with non-linear dependency

We can finally outline a general algorithm for univariate Gaussian inference problems with arbitrary link function $g(x)$. In this case, the likelihood has the following form:

$$p(D \mid x) \propto \prod_{k=1}^{K} \mathcal{N}(y_k; g(x), \sigma_l^2) \ , \tag{31}$$

where σ_l^2 is the likelihood variance. We assume an arbitrary prior Gaussian distribution

$$p(x) = \mathcal{N}(x; \mu_p, \sigma_p^2) \ . \tag{32}$$

Up to constant terms, the (re-parameterized) variational loss is given by

$$\mathcal{L}(\mu, \sigma) \approx \frac{1}{2N} \sum_{n=1}^{N} \left(\sum_{k=1}^{K} (g(x_n) - y_k)^2 / \sigma_l^2 + (x_n - \mu_p)^2 / \sigma_p^2 \right) - \log \sigma \ . \tag{33}$$

where the x_ns are sampled from the variational distribution $q(x; \mu, \sigma)$. Since $\partial x_n / \partial \mu = 1$, the gradient estimator of the mean is

$$\begin{aligned} \frac{\partial}{\partial \mu} \mathcal{L}(\mu, \sigma) &\approx \frac{1}{2N} \sum_{n=1}^{N} \left(\sum_{k=1}^{K} \frac{\partial}{\partial \mu} \left((g(x_n) - y_k)^2 / \sigma_l^2 + (x_n - \mu_p)^2 / \sigma_p^2 \right) \right) \ , \\ &= \frac{1}{N} \sum_{n=1}^{N} \left(\sum_{k=1}^{K} \left((g(x_n) - y_k) g'(x) / \sigma_l^2 + (x_n - \mu_p) / \sigma_p^2 \right) \right) \ . \end{aligned} \tag{34}$$

in this case, we cannot obtain a closed form gradient by taking $N \to \infty$. However, we can use a finite number of samples to approximate the gradient. This results in a stochastic gradient update since the finite sampling introduces an element of randomness in the algorithm. This is similar to the mini-batching used in the training of deep networks. Using this formula $\partial x_n / \partial \sigma = (x_n - \mu)/\sigma$, we obtain

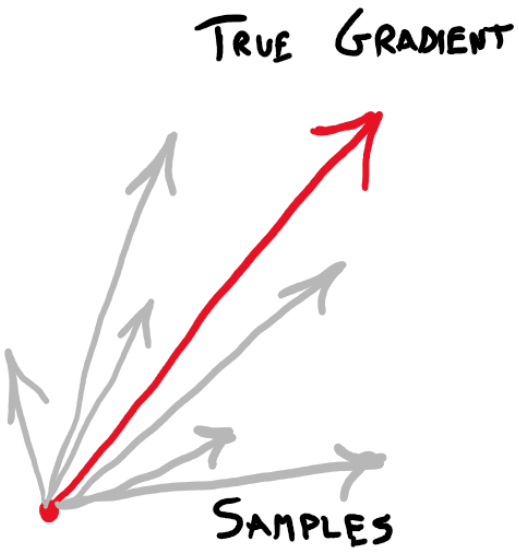

Figure 2: Stochastic gradient directions.

the gradient estimator for the standard deviation:

$$\frac{\partial}{\partial \sigma}\mathcal{L}(\mu,\sigma) \approx \tag{35}$$
$$\frac{1}{\sigma}\frac{1}{N}\sum_{n=1}^{N}\left(\sum_{k=1}^{K}\left((g(x_n)-y_k)g'(x)(x_n-\mu)/\sigma_l^2+(x_n-\mu_p)(x_n-\mu)/\sigma_p^2\right)\right)-\frac{1}{\sigma}\ .$$

Again, this gradient needs to be approximated using a finite number of samples. We can now initialize μ_0 and σ_0 and perform a stochastic gradient update:

$$\mu_{n+1} = \mu_n - \eta\frac{\partial}{\partial \mu}\mathcal{L}^{(N)}(\mu,\sigma)\ , \tag{36}$$

$$\sigma_{n+1} = \sigma_n - \eta\frac{\partial}{\partial \sigma}\mathcal{L}^{(N)}(\mu,\sigma)\ , \tag{37}$$

where $\mathcal{L}^{(N)}(\mu,\sigma)$ denotes a N samples estimate of the gradient. Since the estimate is unbiased, we know that its expectation is equal to the true gradient:

$$\mathbb{E}\left[\nabla\mathcal{L}^{(N)}(\mu,\sigma)\right] = \nabla\mathcal{L}(\mu,\sigma)\ . \tag{38}$$

Furthermore, the variance of the estimator is usually finite and it scales to zero as $N \to \infty$. Under these conditions, it is possible to prove convergence to a local optimum as far as the learning rate η is properly scaled down during training.

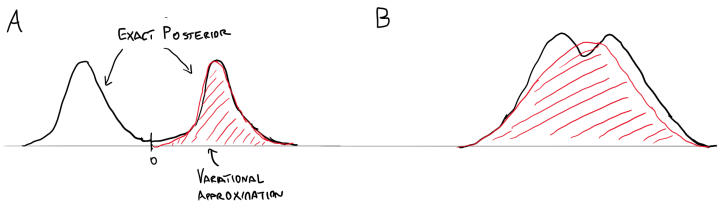

Figure 3: Phase transition in the variational approximation

3.1 Example: Gaussian inference with quadratic link

In the previous example, both the normalization integral and the variational gradient can be obtained in closed form. However, there are several interesting situations where the normalization integral cannot be solved in closed form while the gradient can. For example, consider the inference problem defined by the likelihood

$$p(D \mid x) \propto \prod_{k=1}^{K} \mathcal{N}(y_k; x^2, 1) \ , \tag{39}$$

together with the usual standard Gaussian prior. In this case, the inference cannot be performed in closed form (at least without using complex special functions) since the mean of the observations depend on the latent variable x quadratically instead of linearly. We can obtain the re-parameterization gradient estimator of this new inference problem by using the same technique used in the previous section:

$$\begin{aligned} \frac{\partial}{\partial \mu} \mathcal{L}(\mu, \sigma) &\approx \frac{1}{2N} \sum_{n=1}^{N} \left(\sum_{k=1}^{K} \frac{\partial}{\partial \mu} \left((x_n^2 - y_k)^2 + x_n^2 \right) \right) \ , \\ &= \frac{1}{N} \sum_{n=1}^{N} \left(\sum_{k=1}^{K} (x_n^2 - y_k) 2 x_n + x_n \right) \ , \\ &= -(2K\bar{y} - 1) \sum_{n=1}^{N} x_n + \frac{2K}{N} \sum_{n=1}^{N} x_n^3 \ . \end{aligned} \tag{40}$$

For $N \to \infty$, $\frac{1}{N} \sum_{n=1}^{N} x_n$ again converges to the mean μ while $\frac{1}{N} \sum_{n=1}^{N} x_n^3$ converges to the third *statistical moments* $\mathbb{E}\left[x^2\right]$ and $\mathbb{E}\left[x^3\right]$. Fortunately, all the moments of a Gaussian distribution can be expressed as polynomials in the parameters. In particular:

$$\mathbb{E}_{x \sim q(x; \mu, \sigma^2)}\left[x^3\right] = \mu^3 + 3\mu\sigma^2 \ . \tag{41}$$

This allows us to express the gradient in closed form:

$$\frac{\partial}{\partial \mu} \mathcal{L}(\mu, \sigma) = (-2K\bar{y} + 6K\sigma^2 + 1)\mu + 2K\mu^3 \ . \tag{42}$$

We can obtain a similar closed form formula for the gradient with respect to the variance. If we set this derivative equal to zero, we find either one or three solution depending on the magnitude of the data expectation $\bar{y}$. It is easy to see that $\mu_0 = 0$ is always a solution. The other two possible solutions are

$$\mu_\pm = \pm\sqrt{\bar{y} - 3\sigma^2 - 1/2} \tag{43}$$

which are possible only when $\bar{y} - 3\sigma^2 - 1 \geq 0$.
This behavior can be understood by realizing that the true posterior is bimodal with two peaks at $\pm\sqrt{\bar{y} - 1/2}$ when $\bar{y} - 1/2 \geq 0$ and it is unimodal with a single peak at zero otherwise. This bimodality comes from the symmetry in the likelihood, which gives equal probability to latent values that only differ by their sign. The optima of the variational objective reflects the fact that in the "bimodal phase" we can approximate the posterior with a Gaussian either by fitting the left or the right peak. Note that the location of the variational peek is shifted towards zero if the variational variance is large since in this regime the approximation will attempt to fit both peaks at once in some extent. The μ_0 solution is normally a saddle point, not an optimum. However, when $\bar{y} - 3\sigma^2 - 1 \leq 0$ the separation between the peaks is small compared to the variance and the best approximation suddenly jumps to the origin where both the positive and the negative peak of the true posterior are equally represented.

3.2 Variational inference in discrete models

It is not always possible to obtain a reparameterization estimate of the variational gradient. For example, consider the following model with discrete latents

$$\begin{aligned} p(y \mid x) &= \mathcal{N}(y; x, 1) \\ p(x) &= \text{Binomial}(x; \rho, m) \ , \end{aligned} \tag{44}$$

with the latent variable x following a Bernoulli distribution distribution

$$\text{Bernoulli}(x; \rho) = \rho^x (1-\rho)^(1-x) \ . \tag{45}$$

This is a discrete inference problem with two latent states. Since we only have a small number of states, the posterior is easy to compute in closed form:

$$\begin{aligned} p(x = 1 \mid y) &= \frac{\mathcal{N}(y; 1, 1)\rho}{\mathcal{N}(y; 1, 1)\rho + \mathcal{N}(y; 0, 1)(1-\rho)} \ , \\ &= \frac{1}{1 + \frac{(1-\rho)}{\rho} \exp\left(1/2 - y\right)} \ . \end{aligned} \tag{46}$$

Interestingly, this is just the logistic sigmoid function applied to the argument $(1/2 - y) + \log(1-\rho) - \log\rho$.
Let us now try to solve the same problem using variational gradient descent. First of all, we need to define a parameterized variational distribution. Since this is a two states system, the most general choice is a Bernoulli distribution:

$$q(x; \rho_q) = \rho_q^x (1-\rho_q)^{(1-x)} \ . \tag{47}$$

Note that in this case we know for a fact that our parameterized approximation contains the true posterior for some values of the parameter since all two states distributions can be written as Bernoulli distributions. Up to constant terms that do not depend on the variational parameter ρ_q, the variational loss is

$$\mathcal{L}(\rho_q) = -\mathbb{E}_{x\sim q}\left[-\frac{1}{2}(y-x)^2 + x\log\rho + (1-x)\log(1-\rho)\right] - \mathcal{H}(\rho_q)\ , \quad (48)$$

where $\mathcal{H}(\rho_q)$ is the entropy of a Bernoulli distribution with probability ρ_q:

$$\mathcal{H}(\rho_q) = -\rho_q\log\rho_q - (1-\rho_q)\log(1-\rho_q)\ . \quad (49)$$

In this case, the integral is replaced with a finite sum over the two possible states. We can therefore evaluate the gradient explicitly (check this calculation!)

$$\frac{\partial}{\partial\rho_q}\mathcal{L}(\rho_q) = -\frac{\partial}{\partial\rho_q}\Bigg(\rho_q\left(-\frac{1}{2}(y-1)^2 + \log\rho\right) - (1-\rho_q)\left(-\frac{1}{2}y^2 + \log(1-\rho)\right) \quad (50)$$
$$+ \mathcal{H}(\rho_q)\Bigg) = (1/2-y) - \log\frac{\rho(1-\rho_q)}{(1-\rho)\rho_q}\ .$$

This a a so called *enumerated gradient* since we evaluated all the possibilities explicitly. Not surprisingly, in this case we can obtain the solution in closed form by setting the derivative equal to zero (try it!).

3.3 Stochastic gradients without reparameterizations

As you can imagine, enumeration gradients become intractable when the number of states is large. It would be therefore handy to have a a stochastic gradient estimator similar to the reparameterization gradients we used in the previous sections. Remember that the reparameterization estimator was obtained by replacing the expectation integral in the variational loss with a finite average and then taking the gradient of the resulting finite expression. In the case of the Bernoulli model, this gives

$$\frac{\partial}{\partial\rho_q}\mathcal{L}(p_q) \approx \frac{\partial}{\partial\rho_q}\frac{1}{N}\sum_{n=1}^{N}\left(\frac{1}{2}(y-x_n)^2 - x_n\log\rho - (1-x_n)\log(1-\rho)\right) - \frac{\partial}{\partial\rho_q}\mathcal{H}(\rho_q) \quad (51)$$
$$= \frac{1}{N}\sum_{n=1}^{N}\left((y-x_n) - \log\rho - \log(1-\rho)\right)\frac{\partial x_n}{\partial\rho_q} - \frac{\partial}{\partial\rho_q}\mathcal{H}(\rho_q)\ .$$

Unfortunately, a sample from a Bernoulli distribution cannot be written as a differentiable function of the parameter since x_n is either 0 or 1 and therefore it cannot vary smoothly as ρ_q changes. Therefore, the derivative $\frac{\partial x_n}{\partial\rho_q}$ does not exist and this expression is a no-go. Is is still possible to obtain a stochastic gradient estimator? Yes. The trick is to compute the exact gradient first as an

expectation and then approximate the resulting expression as a finite average. We can start by computing the exact derivative:

$$\frac{\partial}{\partial \rho_q}\mathcal{L}(\rho_q) = -\frac{\partial}{\partial \rho_q}\sum_{x=0,1} q(x)\log p(y \mid x)p(x) - \frac{\partial}{\partial \rho_q}\mathcal{H}(\rho_q) \tag{52}$$
$$- \sum_{x=0,1}\frac{\partial q(x;\rho_q)}{\partial \rho_q}\log p(y \mid x)p(x) - \frac{\partial}{\partial \rho_q}\mathcal{H}(\rho_q)\ .$$

The entropy term is not problematic as we already evaluated it in closed form. We now need to re-express the sum term as an expectation with respect to q. This can be done by using the formula

$$\frac{\partial}{\partial \rho_q}\log q(x;\rho_q) = \frac{\frac{\partial q(x;\rho_q)}{\partial \rho_q}}{q(x;\rho_q)}\ , \tag{53}$$

which implies that

$$\frac{\partial q(x;\rho_q)}{\partial \rho_q} = q(x;\rho_q)\frac{\partial}{\partial \rho_q}\log q(x;\rho_q)\ . \tag{54}$$

We can now use this formula in Eq. 52:

$$\frac{\partial}{\partial \rho_q}\mathcal{L}(\rho_q) = -\sum_{x=0,1} q(x;\rho_q)\left(\frac{\partial}{\partial \rho_q}\log q(x;\rho_q)\right)\log p(y \mid x)p(x) - \frac{\partial}{\partial \rho_q}\mathcal{H}(\rho_q) \tag{55}$$
$$= \mathbb{E}_{x\sim q(x;\rho_q)}\left[\left(\frac{\partial}{\partial \rho_q}\log q(x;\rho_q)\right)\log p(y \mid x)p(x)\right] - \frac{\partial}{\partial \rho_q}\mathcal{H}(\rho_q)$$
$$= \mathbb{E}_{x\sim q(x;\rho_q)}\left[\left(\frac{x}{\rho_q} + \frac{(1-x)}{(1-\rho_q)}\right)\log p(y \mid x)p(x)\right] - \frac{\partial}{\partial \rho_q}\mathcal{H}(\rho_q)\ .$$

As you can see, the gradient is now in the form of an expectation with respect to q, which does not involve any undefined derivative. This is possible because the log-probability $\log q(x;\rho_q)$ is a differentiable function of ρ_q. We can now replace the exact expectation with a finite average. The result is the *reinforce gradient estimator*:

$$\frac{\partial}{\partial \rho_q}\mathcal{L}(\rho_q) \approx \frac{1}{N}\sum_{n=1}^{N}\left(\frac{\partial}{\partial \rho_q}\log q(x_n;\rho_q)\right)\log p(y \mid x_n)p(x_n) - \frac{\partial}{\partial \rho_q}\mathcal{H}(\rho_q)\ . \tag{56}$$

Note that this estimator does not require the (non-existent) derivative of the sample with respect to the variational parameter. While the estimator is again unbiased, it unfortunately tends to have high variance, which often leads to very slow convergence.