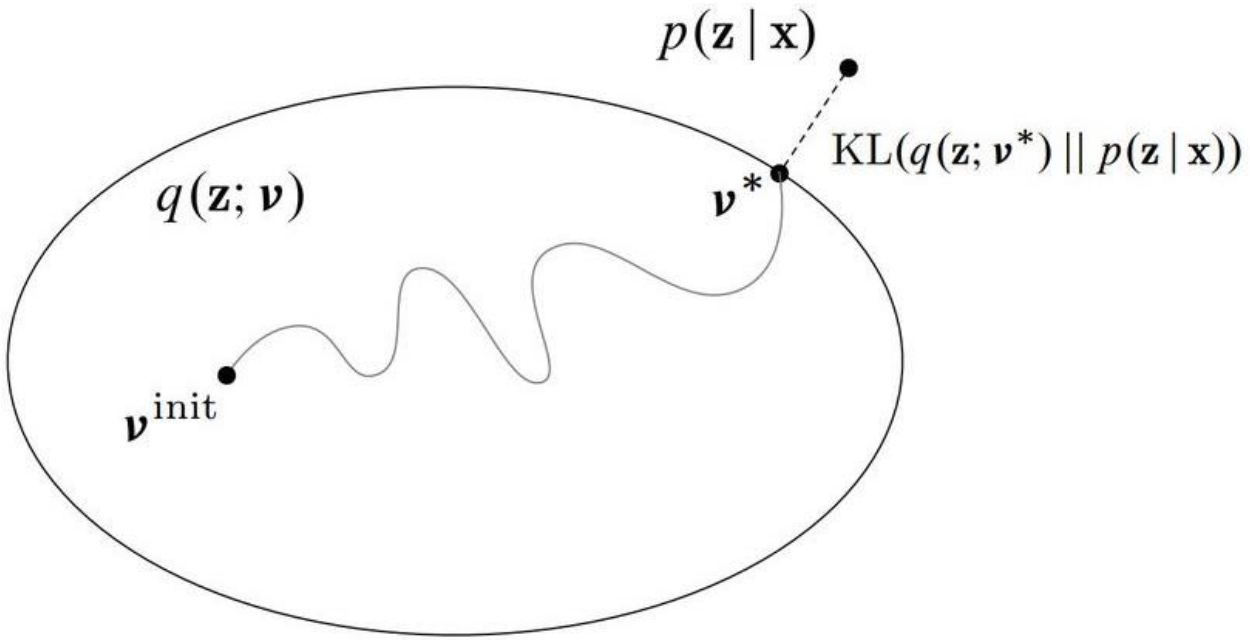
Variational inference



Marginal likelihood

- Assume you have statistics $p(x, z)$ of eye colors x per nationality z

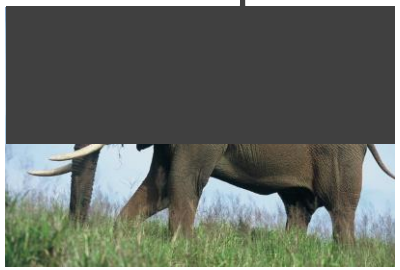
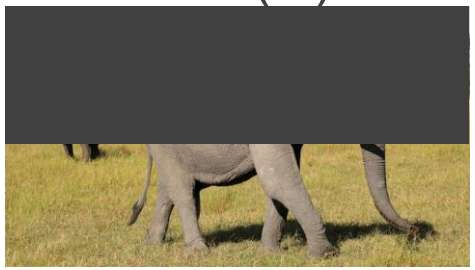
	Dutch	Greek	Chinese	Indian	Italian	German	US	Spanish
Brown	0.02	0.03	0.02	0.01	0.07	0.03	0.01	0.00
Blue	0.09	0.01	0.03	0.03	0.08	0.04	0.03	0.06
Green	0.01	0.01	0.08	0.06	0.07	0.08	0.06	0.06

- If we want to know the distribution of one of our variables (*e.g.*, eye colors)
  - → we sum up over all possible outcomes (marginalize) of the other variable
  - *E.g.*, nationalities for the marginal likelihood $p(x) = \sum_z p(x, z)$

Color	
Brown	0.19
Blue	0.37
Green	0.44
Total	1.00

Marginal likelihood

- Or assume that our bottom half pixels are visible ($\boldsymbol{x}$) and the upper half not ($\boldsymbol{z}$)

- Assume we somehow know a good model $p(\boldsymbol{x}, \boldsymbol{z})$ of how the visible and hidden/latent pixels interact
- Let's say we want to know how likely bottom half the image is to be observed
- We must marginalize out all possible latents $\boldsymbol{z}$ to fill the rest of the image
  - For instance, some pictures might contain one, two, or more elephants

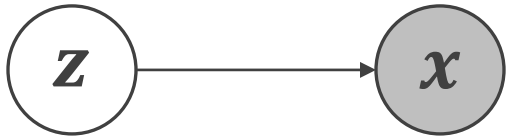
$$p(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z})$$

Marginal likelihood in latent variable models

- When "learning to represent" an input x we assume a latent variable z
  - and try to explain $\boldsymbol{x}$ using all possible $\boldsymbol{z}$

$$p(\boldsymbol{x}) = \int_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z})\, d\boldsymbol{z} = \int_{\boldsymbol{z}} p(\boldsymbol{x}|\boldsymbol{z})\, p(\boldsymbol{z}) d\boldsymbol{z} = \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{p}(\boldsymbol{z})}[p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})]$$

- Hence, a latent variable model can be viewed as a generation process
  - First, we generate a new $\boldsymbol{z}$ from $p(\boldsymbol{z})$ by sampling
  - Then, we generate a new $\boldsymbol{x}$ by sampling from the $p(\boldsymbol{x}|\boldsymbol{z})$ given the sampled $\boldsymbol{z}$



Intractable $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$

- Question: how to find the optimal parameters $\boldsymbol{\theta}$?
- Maximizing log-likelihood, again

$$\log \prod_{\boldsymbol{x} \in D} p(\boldsymbol{x}) = \sum_{\boldsymbol{x}} \log p(\boldsymbol{x}) = \sum_{\boldsymbol{x}} \log \sum_{\boldsymbol{z}} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$$

- Like in Boltzmann machines, the $\sum_{\boldsymbol{z}} \ldots$ is a nasty one
  - E.g., for a 3-dimensional binary $\boldsymbol{z}$ iterate over [0,0,0],[0,0,1],[0,1,1],…
  - For 20 dimensions $2^{20} \approx 1M$ latents and generations. Per image $\boldsymbol{x}$!
  - For continuous $\boldsymbol{z}$ even harder, we cannot even enumerate

Making $p_{\theta}(x, z)$ tractable with naive Monte Carlo

- We want to optimize per data point x: $\log \sum_z p_{\theta}(x, z)$
- The sum contains a bunch of probabilities
  - equivalent to expected value times the number of summands

$$\log \sum_z p_{\theta}(x, z) = \log |Z| \, \mathbb{E}[p_{\theta}(x, z)]$$

- Do we need all the summands to compute the expected value (average)
  - No, if we sample randomly z (uniformly) and average, it gives us an estimate
  - Basically replace whole sum with a weighted smaller sum

$$\log |Z| \, \mathbb{E}[p_{\theta}(x, z)] \approx \log \frac{|Z|}{K} \mathbb{E}^{(K)}_{z \sim \text{Uniform}}[p_{\theta}(x, z)] = \log \frac{|Z|}{K} \sum_{k=1}^{K} p_{\theta}(x, z_k)$$

- Doesn't scale, too many samples for the expectation estimate to be accurate
  - Most z_k would be in 'very low density regions' → Unimportant $p_{\theta}(x, z_k)$
  - In technical terms, this is a 'high variance' estimator

Making $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$ tractable with importance sampling MC

- Better if select few good summands in the sum $\sum_{k=1}^{K} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}_k)$
- If, theoretically, we had a nice distribution around the mass of relevant $\boldsymbol{z}_k$
  - we could use that distribution to sample $\boldsymbol{z}_k$ and get a better sample average with fewer k

$$\log \sum_{\boldsymbol{z}} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = \log \sum_{\boldsymbol{z}} q_{\varphi}(\boldsymbol{z}) \frac{1}{q_{\varphi}(\boldsymbol{z})} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$$

$$= \log \mathbb{E}_{\boldsymbol{z} \sim q_{\varphi}(\boldsymbol{z})} \left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{q_{\varphi}(\boldsymbol{z})} \right] \approx \log \frac{1}{K} \sum_{k=1}^{K} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}_k)}{q_{\varphi}(\boldsymbol{z}_k)}, \text{ where } \boldsymbol{z}_k \text{ are sampled from } q_{\varphi}(\boldsymbol{z})$$

- Note the dual use of $q_{\varphi}(\boldsymbol{z})$
  - In the nominator $q_{\varphi}(\boldsymbol{z})$ is the density function we use as sampling mechanism. By sampling from it (e.g., Gaussian samples if it is Gaussian) this quantity is used and disappears by the sum
  - In the denominator $q_{\varphi}(\boldsymbol{z}_k)$ is simply a function. We feed it $\boldsymbol{z}_k$ and returns how important $\boldsymbol{z}_k$ is for our probability space
- Scales much better and with much lower variance, but we don't know what is a good $q_{\varphi}(\boldsymbol{z}_k)$

Learning the importance sampling distribution

- Importance sampling is promising but how to determine $q_{\varphi}(\boldsymbol{z}_k)$?
- Learn $q_{\varphi}(\boldsymbol{z}_k)$ from data!
- Our learning objective is to maximize the log probability

$$\log \mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right] \approx \log \frac{1}{K}\sum_{k=1}^{K}\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z}_k)}{q_{\boldsymbol{\varphi}}(\boldsymbol{z}_k)}$$

- The log $\mathbb{E}$ stands for logarithm of an unknown integral
  - not very convenient for derivations and computations
- Would be much nicer if we could swap the log $\mathbb{E}$ to $\mathbb{E}$ log
  - Then we would simply need the expectation of the logarithm of a function
  - Especially convenient if $p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})$ belongs to the exponential family

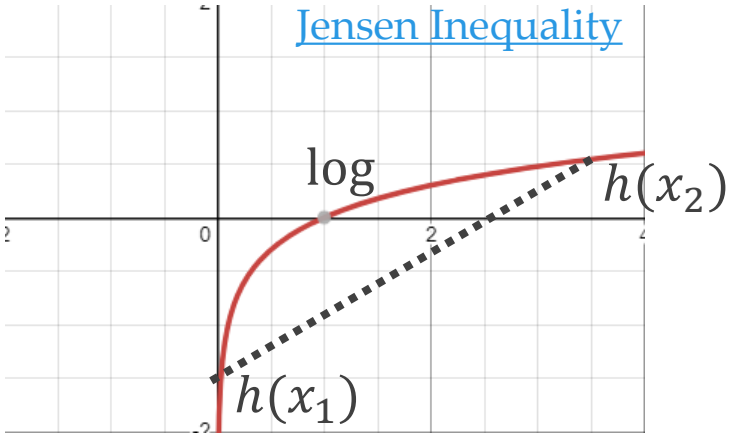
Jensen's inequality

- A concave function h (like a logarithm)on a sum will always be larger than the sum of h on individual summands
  - Basically, a line connecting two points of a function will be always below the function

$$h(tx_1 + (1-t)x_2) \geq th(x_1) + (1-t)h(x_2)$$

- With probabilities and random variables this translates to

$$h(\mathbb{E}[\boldsymbol{x}]) \geq \mathbb{E}[h(\boldsymbol{x})]$$



A lower bound on the maximum likelihood

- By applying Jensen's inequality

$$\log \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{q_\varphi}(\boldsymbol{z})}\left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right] \geq \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{q_\varphi}(\boldsymbol{z})}\left[\log\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right]$$

- We replaced the original ML objective with a quantity that is always smaller
  - (1) By improving $\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{q_\varphi}(\boldsymbol{z})}\left[\log\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right]$ we always improve $\log \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{q_\varphi}(\boldsymbol{z})}\left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right]$
  - 'Lower bound'
- (2) $\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{q_\varphi}(\boldsymbol{z})}\left[\log\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right]$ is a tractable & comfortable quantity → easy optimization
  - An expectation → Monte Carlo sampling is possible
  - The log can couple nicely with $p_{\boldsymbol{\theta}}$ if chosen properly

Making $p(\boldsymbol{z}|\boldsymbol{x})$ tractable with variational inference

- We can also view variational inference from the lens of intractability
- The problematic quantity in our latent model is the posterior
  - The reason is the intractable normalization

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p(\boldsymbol{x},\boldsymbol{z})}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x},\boldsymbol{z})}{\int \boldsymbol{p(x,z)d\boldsymbol{z}}}$$

- Variational inference approximates the true posterior $p(\boldsymbol{z}|\boldsymbol{x})$ with $q_\varphi(\boldsymbol{z}|\boldsymbol{x})$

$$\mathrm{KL}(q(\boldsymbol{z}) \parallel p(\boldsymbol{z}|\boldsymbol{x})) = \int q_\varphi(\boldsymbol{z}) \log \frac{q_\varphi(\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z}$$

$$= -\int q_\varphi(\boldsymbol{z}) \log \frac{p(\boldsymbol{x},\boldsymbol{z})}{p(\boldsymbol{x})q_\varphi(\boldsymbol{z})} d\boldsymbol{z} = -\int q_\varphi(\boldsymbol{z}) \log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\varphi(\boldsymbol{z})} + \int q_\varphi(\boldsymbol{z}) \log p(\boldsymbol{x})\, d\boldsymbol{z}$$

$$= -\mathbb{E}_{q_\varphi(\boldsymbol{z})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\varphi(\boldsymbol{z})}\right] + \log p(\boldsymbol{x})$$

Evidence lower bound

- $\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{q}_{\varphi}(\boldsymbol{z})}\left[\log\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\varphi}(\boldsymbol{z})}\right]$ also known as 'evidence lower bound'

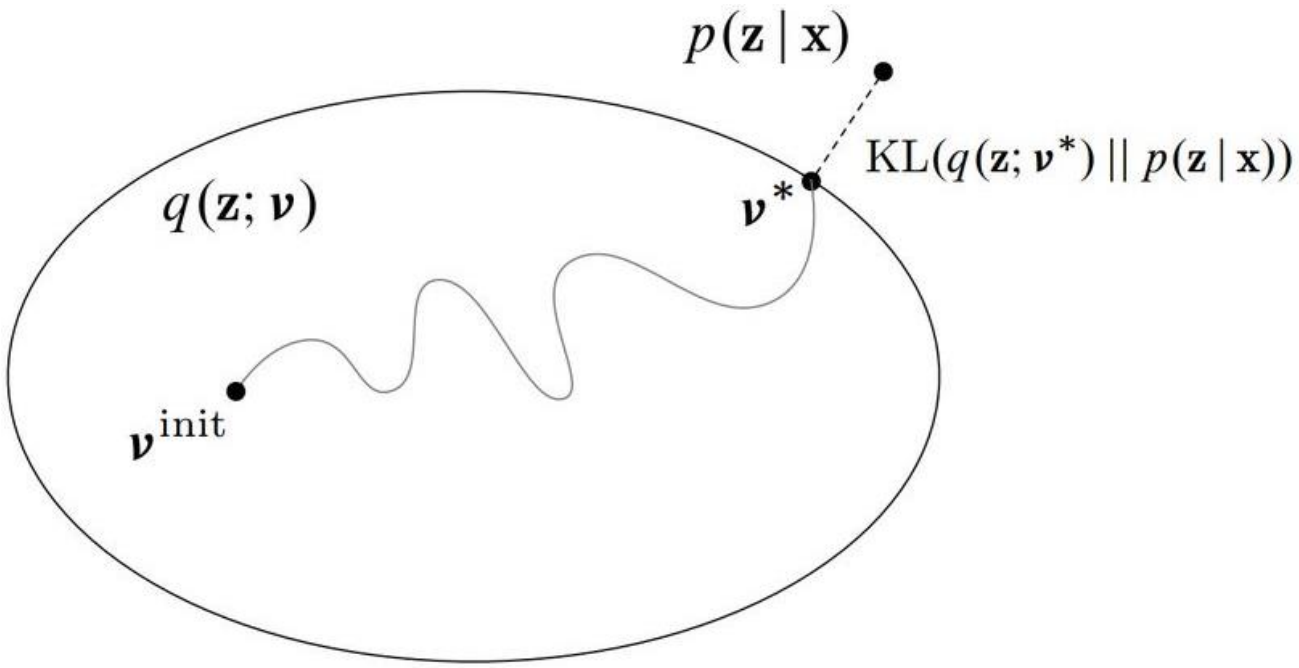
$$\log p(\boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{z})}\left[\log\frac{p(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right] + \mathrm{KL}\big(q_{\boldsymbol{\varphi}}(\boldsymbol{z}) \parallel p(\boldsymbol{z}|\boldsymbol{x})\big)$$
$$= \mathrm{ELBO} + \mathrm{KL}\big(q_{\boldsymbol{\varphi}}(\boldsymbol{z}) \parallel p(\boldsymbol{z}|\boldsymbol{x})\big)$$

- Why 'evidence'?
  - The KL term is always positive
  - If we drop it, we bound the log evidence $\log p(\boldsymbol{x})$ from below

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\left[\log\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right]$$

- Higher ELBO → smaller difference to true $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ → better latent representation
- Higher ELBO → gap to log-likelihood tightens → better density model

Variational inference, graphically



ELBO balancing reconstruction and the prior

- We can expand the ELBO as

$$\mathbb{E}_{q(\boldsymbol{z})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right] = \mathbb{E}_{q(\boldsymbol{z})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right]$$
$$= \mathbb{E}_{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}\left[q_{\boldsymbol{\varphi}}(\boldsymbol{z}) \parallel p(\boldsymbol{z})\right]$$

- The first term encourages the reconstructions that the maximize likelihood
- The second term minimizes the distance of the variational distribution from the prior

ELBO and entropy regularization

- We can also expand the ELBO as

$$\mathbb{E}_{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right] = \mathbb{E}_{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}\right]$$
$$= \mathbb{E}_{q(\boldsymbol{z})}[\log p(\boldsymbol{x},\boldsymbol{z})] - \mathbb{E}_{q(\boldsymbol{z})}[\log q_{\boldsymbol{\varphi}}(\boldsymbol{z})]$$
$$= \mathbb{E}_{q(\boldsymbol{z})}[\log p(\boldsymbol{x},\boldsymbol{z})] + H(q_{\boldsymbol{\varphi}}(\boldsymbol{z}))$$

where H(·) is the entropy

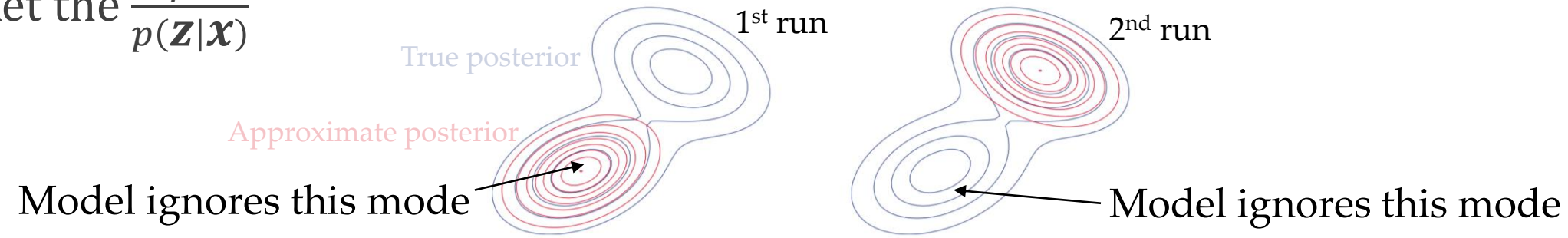
- Maximizing the joint likelihood → Something like the Boltzmann energy
- While maintaining enough entropy ('uncertainty') in the distribution of latents
  - Avoiding latents to collapse to pathological, point estimates ($\boldsymbol{z}$ as single values)

Variational inference underestimates variance

- If you noticed, for the second way to derive the ELBO we minimized

$$\mathrm{KL}\big(q_{\boldsymbol{\varphi}}(\boldsymbol{z}) \parallel p(\boldsymbol{z}|\boldsymbol{x})\big) = \int q_{\boldsymbol{\varphi}}(\boldsymbol{z}) \log \frac{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z}$$

  - We want to sample from $q_{\boldsymbol{\varphi}}(\boldsymbol{z})$ in expectations later on, as $p(\boldsymbol{z}|\boldsymbol{x})$ is intractable
- The model wants to approximate $p(\boldsymbol{z}|\boldsymbol{x}) \rightarrow$ can't really know where $p(\boldsymbol{z}|\boldsymbol{x})$ is low
- The model prefers to hedge and 'bias' $q_{\boldsymbol{\varphi}}(\boldsymbol{z})$ towards 0 for regions it can't be certain
  - Better pick one mode (randomly) than miss a 'zero' density region of $p(\boldsymbol{z}|\boldsymbol{x})$ and skyrocket the $\frac{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})}$



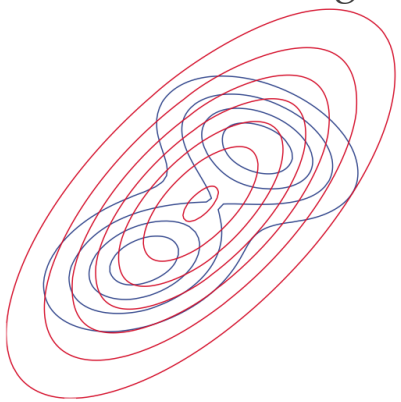
How to overestimate variance?

- You would need to use the forward KL

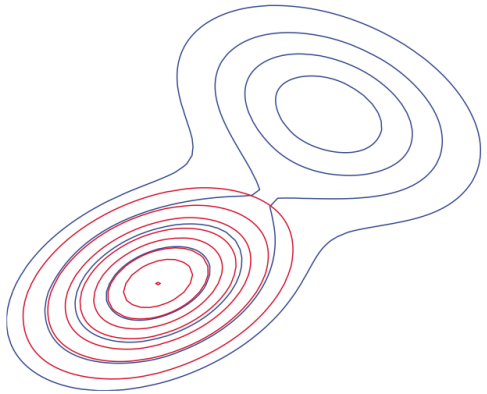
$$\mathrm{KL}\big(p(\boldsymbol{z}|\boldsymbol{x}) \parallel q_{\boldsymbol{\varphi}}(\boldsymbol{z})\big) = \int p(\boldsymbol{z}|\boldsymbol{x}) \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})} d\boldsymbol{z}$$

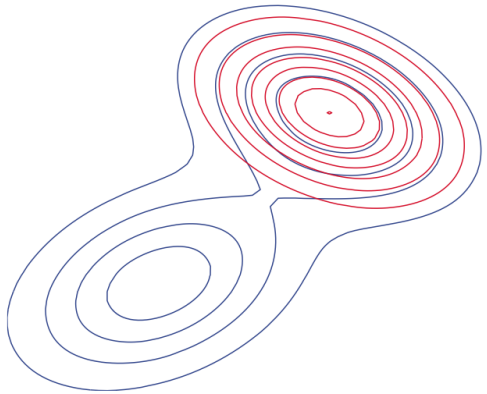
- The model would prefer placing some density everywhere
  - That way it avoids $\frac{p(\boldsymbol{z}|\boldsymbol{x})}{q_{\boldsymbol{\varphi}}(\boldsymbol{z})}$ skyrocketing if it misses areas where $p(\boldsymbol{z}|\boldsymbol{x})$

Overestimating variance



Underestimating variance





Variational Inference and Variational Autoencoders

- Variational Inference is a general machine learning methodology
  - Not specific to Variational Autoencoders
- In the original Variational Inference, we have $q_{\boldsymbol{\phi}}(\boldsymbol{z}) = q(\boldsymbol{z}; \boldsymbol{\varphi})$ where $\boldsymbol{\varphi} = \{\boldsymbol{\varphi_i}\}$ per data point
  - Not $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, this is specific in Variational Autoencoders
  - That is, we optimize a different neural network $\boldsymbol{\varphi_i}$ per data point
  - This is inefficient and cannot generalize ($\boldsymbol{\varphi_i}$ are only for the training)
- As we will see, Variational Autoencoders share the parameters $\boldsymbol{\varphi}$ among all data points (single neural network in the end)
  - This is called amortization
  - And introduce specialization by conditioning on the data point $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$