# Variational Inference and Learning II

## Modern VI and Variational Autoencoders

Michael U. Gutmann

Probabilistic Modelling and Reasoning (INFR11134)
School of Informatics, The University of Edinburgh

Autumn Semester 2025

# Assumptions

- Model: $p(\mathbf{v}, \mathbf{h}; \theta)$
- Data: $\mathcal{D} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$, $\mathbf{v}_i \overset{\text{iid}}{\sim} p_*$
- The model is a latent variable model: we have observations for all dimensions of $\mathbf{v}$ but no observations of the latents $\mathbf{h}$.
- For each observation $\mathbf{v}_i$, there is a latent $\mathbf{h}_i$.
- Because of iid assumption,

$$p(\mathbf{v}_1, \ldots, \mathbf{v}_n, \mathbf{h}_1, \ldots, \mathbf{h}_n; \theta) = \prod_{i=1}^{n} p(\mathbf{v}_i, \mathbf{h}_i; \theta) \qquad (1)$$

- We do not deal with the case of unobserved variables due to missing data, i.e. incomplete observations of $\mathbf{v}$. (For VI work on this topic, see e.g. Simkus et al, *Variational Gibbs Inference for Statistical Model Estimation from Incomplete Data*, Journal of Machine Learning Research, 2023)

# Program

1. Scalable generic variational learning of latent variable models

2. Deep latent variable models and variational autoencoders

# Program

1. Scalable generic variational learning of latent variable models
   - ELBO for iid data
   - Amortised variational inference
   - Reparameterisation and stochastic optimisation

2. Deep latent variable models and variational autoencoders

# Lower bound on the likelihood for iid data

- We had

$$\mathcal{L}_{\mathbf{x}}(q) = \mathbb{E}_{q(\mathbf{y}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{y})}{q(\mathbf{y}|\mathbf{x})}\right] \tag{2}$$

- Substitute

$$\mathbf{x} \to (\mathbf{v}_1, \ldots, \mathbf{v}_n) \qquad p(\mathbf{x},\mathbf{y}) \to \prod_{i=1}^{n} p(\mathbf{v}_i, \mathbf{h}_i; \boldsymbol{\theta}) \tag{3}$$

$$\mathbf{y} \to (\mathbf{h}_1, \ldots, \mathbf{h}_n) \tag{4}$$

- Since the true conditional factorises, we use

$$q(\mathbf{h}_1, \ldots, \mathbf{h}_n|\mathbf{v}_1, \ldots, \mathbf{v}_n) = \prod_{i=1}^{n} q(\mathbf{h}_i|\mathbf{v}_i) \tag{5}$$

- We have one conditional variational distribution $q(\mathbf{h}|\mathbf{v})$.

# Lower bound on the likelihood for iid data

- The ELBO $\mathcal{L}_{\mathcal{D}}$ for iid data $\mathcal{D} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ becomes a sum of per data-point ELBOs $\mathcal{L}_{\mathbf{v}_i}$, denoted by $\mathcal{L}_i$:

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, q) = \sum_{i=1}^{n} \mathcal{L}_i(\boldsymbol{\theta}, q) \tag{6}$$

$$\mathcal{L}_i(\boldsymbol{\theta}, q) = \mathbb{E}_{q(\mathbf{h}_i|\mathbf{v}_i)}\left[\log \frac{p(\mathbf{v}_i, \mathbf{h}_i; \boldsymbol{\theta})}{q(\mathbf{h}_i|\mathbf{v}_i)}\right] \tag{7}$$

- From the basic properties of the ELBO, we have

$$\mathcal{L}_i(\boldsymbol{\theta}, q) = \log p(\mathbf{v}_i; \boldsymbol{\theta}) - \mathrm{KL}(q(\mathbf{h}_i|\mathbf{v}_i)||p(\mathbf{h}_i|\mathbf{v}_i; \boldsymbol{\theta})) \tag{8}$$

- This gives

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, q) = \sum_{i=1}^{n} [\log p(\mathbf{v}_i; \boldsymbol{\theta}) - \mathrm{KL}(q(\mathbf{h}_i|\mathbf{v}_i)||p(\mathbf{h}_i|\mathbf{v}_i; \boldsymbol{\theta}))] \tag{9}$$

# Lower bound on the likelihood for iid data

- With $\ell(\boldsymbol{\theta}) = \sum_i \log p(\mathbf{v}_i; \boldsymbol{\theta})$ we obtain

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, q) = \ell(\boldsymbol{\theta}) - \sum_{i=1}^{n} \mathrm{KL}(q(\mathbf{h}_i|\mathbf{v}_i)||p(\mathbf{h}_i|\mathbf{v}_i; \boldsymbol{\theta})) \tag{10}$$

- Maximum likelihood estimation

$$\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}, q} \mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, q) \tag{11}$$

# Key technical difficulties

- We have to maximise $\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, q) = \sum_{i=1}^{n} \mathcal{L}_i(\boldsymbol{\theta}, q)$ with respect to $\boldsymbol{\theta}$ and the conditional $q(\mathbf{h}|\mathbf{v})$.
- We had

$$\mathcal{L}_i(\boldsymbol{\theta}, q) = \mathbb{E}_{q(\mathbf{h}_i|\mathbf{v}_i)} \left[ \log \frac{p(\mathbf{v}_i, \mathbf{h}_i; \boldsymbol{\theta})}{q(\mathbf{h}_i|\mathbf{v}_i)} \right] \tag{12}$$

  Analytical closed form expression only available in special cases.
- We do not want to restrict the model class but solve the optimisation problem for large $n$ and generic $p(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta})$.
- Key technical difficulties are:
  1. Learning of conditional variational distribution $q(\mathbf{h}|\mathbf{v})$
  2. Maximisation when the objective involves the $\mathbb{E}_{q(\mathbf{h}_i|\mathbf{v}_i)}$

# Issue 1: Learning the conditional variational distribution

- Learning the conditional $q(\mathbf{h}|\mathbf{v})$ is hard since we have to effectively learn infinitely many pdfs/pmfs (one for each $\mathbf{v}$!).
- $\mathcal{L}_i$ only involves $q(\mathbf{h}_i|\mathbf{v}_i)$. Hence we could optimise $\mathcal{L}_\mathcal{D}$ by optimising each $\mathcal{L}_i$ with respect to $q_i(\mathbf{h}_i) = q(\mathbf{h}_i|\mathbf{v}_i)$

$$\max_q \mathcal{L}_\mathcal{D}(\boldsymbol{\theta}, q) \Leftrightarrow \max_{q_i} \mathcal{L}_i(\boldsymbol{\theta}, q_i) \quad \text{for } i = 1, \ldots, n \tag{13}$$

- We typically make some parametric assumptions. Let $q_i(\mathbf{h})$ be parameterised as $q_i(\mathbf{h}; \boldsymbol{\lambda}_i) \in \mathcal{Q}_i$.
- Different $q_i(\mathbf{h}; \boldsymbol{\lambda}_i)$ may belong to different parametric families.
- Optimisation with respect to $q_i$ then becomes optimisation with respect to $\boldsymbol{\lambda}_i$.

# Issue 1: Learning the conditional variational distribution

- Closed form solution typically not available. This means that we have to iteratively optimise $\mathcal{L}_i$ with respect to $\boldsymbol{\lambda}_i$ for all data points.
- Feasible if $n$ is very small. But too costly otherwise.

# Amortisation

- Let us parameterise the conditional distribution $q(\mathbf{h}|\mathbf{v})$ directly as

$$q(\mathbf{h}|\mathbf{v}) = q_\phi(\mathbf{h}|\mathbf{v}) = q(\mathbf{h}; \boldsymbol{\lambda}_\phi(\mathbf{v})) \tag{14}$$

  where $\boldsymbol{\lambda}_\phi(\mathbf{v})$ is a nonlinear function parameterised by $\phi$. It is called inference or encoder network, or simply encoder.
- This means that we assume that each $q(\mathbf{h}|\mathbf{v})$ belongs to the same parametric family $\mathcal{Q} = \{q(\mathbf{h}; \boldsymbol{\lambda})\}_\lambda$.
- The function $\boldsymbol{\lambda}_\phi(\mathbf{v})$ maps each $\mathbf{v}$ to its corresponding parameter value $\boldsymbol{\lambda}$.
- Note: $\boldsymbol{\lambda}$ are the parameters of the variational distribution while $\phi$ are the parameters of the encoder network.
- Denote $\mathcal{L}_i(\boldsymbol{\theta}, q_\phi)$ by $\mathcal{L}_i(\boldsymbol{\theta}, \phi)$ and $\mathcal{L}_\mathcal{D}(\boldsymbol{\theta}, q_\phi)$ by $\mathcal{L}_\mathcal{D}(\boldsymbol{\theta}, \phi)$.
- We learn $\phi$ by maximising

$$\mathcal{L}_\mathcal{D}(\boldsymbol{\theta}, \phi) = \sum_{i=1}^{n} \mathcal{L}_i(\boldsymbol{\theta}, \phi) \tag{15}$$

# Amortisation (example)

- A popular choice for $q_\phi(\mathbf{h}|\mathbf{v})$ is

$$q_\phi(\mathbf{h}|\mathbf{v}) = \prod_k^H q_\phi(h_k|\mathbf{v}) \tag{16}$$

$$q_\phi(h_k|\mathbf{v}) = \mathcal{N}(h_k; \mu_k(\mathbf{v}; \phi_k^\mu), \sigma_k^2(\mathbf{v}; \phi_k^\sigma) \tag{17}$$

  $\phi$ denotes parameters needed to parameterise all mean and var functions.
- Often used for variational autoencoders (see later).
- Makes both an independence and parametric assumption.
- This means that $\mathcal{Q} = \{q(\mathbf{h}; \boldsymbol{\lambda})\}_\lambda$ equals the factorised Gaussian family with parameters

$$\boldsymbol{\lambda} = (\mu_1, \ldots, \mu_H, \sigma_1^2, \ldots, \sigma_H^2) \tag{18}$$

- The mapping $\boldsymbol{\lambda}_\phi(\mathbf{v})$ maps $\mathbf{v}$ to the means and variances,

$$(\mu_1, \ldots, \mu_H, \sigma_1^2, \ldots, \sigma_H^2) = \boldsymbol{\lambda}_\phi(\mathbf{v}) \tag{19}$$

# Amortisation gap

- $\mathcal{L}_{\mathcal{D}}$ is maximised if all individual per data-point $\mathcal{L}_i$ are maximised.
- When learning $\phi$, we hope that after learning

$$q(\mathbf{h}_i; \boldsymbol{\lambda}_{\hat{\phi}}(\mathbf{v}_i)) \approx \operatorname*{argmax}_{q_i \in \mathcal{Q}_i} \mathcal{L}_i(\boldsymbol{\theta}, q_i) \quad \text{for all } i \tag{20}$$

- The optimisation $\operatorname{argmax}_{q_i} \mathcal{L}_i$ maps $\mathbf{v}_i$ to the optimal $q_i$, and the idea of amortised inference is to approximate this mapping.
- However, the approximation will not be perfect because
  - $\boldsymbol{\lambda}_{\phi}(\mathbf{v})$ is learned by maximising the sum $\sum_i \mathcal{L}_i(\boldsymbol{\theta}, \phi)$ and not a single $\mathcal{L}_i(\boldsymbol{\theta}, \phi)$ for a given $\mathbf{v}_i$.
  - We assume that all $q(\mathbf{h}|\mathbf{v})$ belong to the same parametric family, i.e. $\mathcal{Q} = \mathcal{Q}_i$ for all $i$, which may not be the case.
- The approximation will be better for some $\mathbf{v}_i$ than for others.

# Amortisation gap

- The approximation error due to amortisation is

$$q_i^*(\mathbf{h}_i|\mathbf{v}_i) - q(\mathbf{h}_i; \boldsymbol{\lambda}_{\hat{\phi}}(\mathbf{v}_i)), \quad q_i^*(\mathbf{h}_i|\mathbf{v}_i) = \operatorname*{argmax}_{q_i \in \mathcal{Q}_i} \mathcal{L}_i(\boldsymbol{\theta}, q_i) \quad (21)$$

  (If $\mathcal{Q} = \mathcal{Q}_i$, we can also compare the amortised with the optimal parameter $\boldsymbol{\lambda}$)

- Difference between corresponding ELBOs is called the amortisation gap

$$\mathcal{L}_i(\boldsymbol{\theta}, q_i^*) - \mathcal{L}_i(\boldsymbol{\theta}, \hat{\phi}) \quad \text{with } \hat{\phi} = \operatorname*{argmax}_{\phi} \mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, \phi) \quad (22)$$

- After learning, the encoder network $\boldsymbol{\lambda}_{\hat{\phi}}(\mathbf{v})$ can be applied to test inputs $\mathbf{v}_{\text{test}}$ thereby bypassing an optimisation of the ELBO $\mathcal{L}_{\mathbf{v}_{\text{test}}}$.
- The approximation error and amortisation gap will likely be larger for $\mathbf{v}_{\text{test}}$ than for the training data $\mathbf{v}_1, \ldots, \mathbf{v}_n$.

  For methods to reduce the amortisation gap, see e.g. Marino et al, *Iterative amortised inference*, ICML 2018, `https://arxiv.org/abs/1807.09356`

# Amortisation gap

- Example in two dimensions where $q_i$ is assumed Gaussian with parameters $\boldsymbol{\lambda} = (\mu_1, \mu_2)$.
- The contour plot shows $\mathcal{L}_i(\boldsymbol{\theta}, q_i)$ as a function of $\boldsymbol{\lambda}$
- The blue line shows the gradient ascent optimisation path when the ELBO is optimised without amortisation.
- The cyan diamond shows the amortised estimate $\boldsymbol{\lambda}_{\hat{\phi}}(\mathbf{v}_i)$.

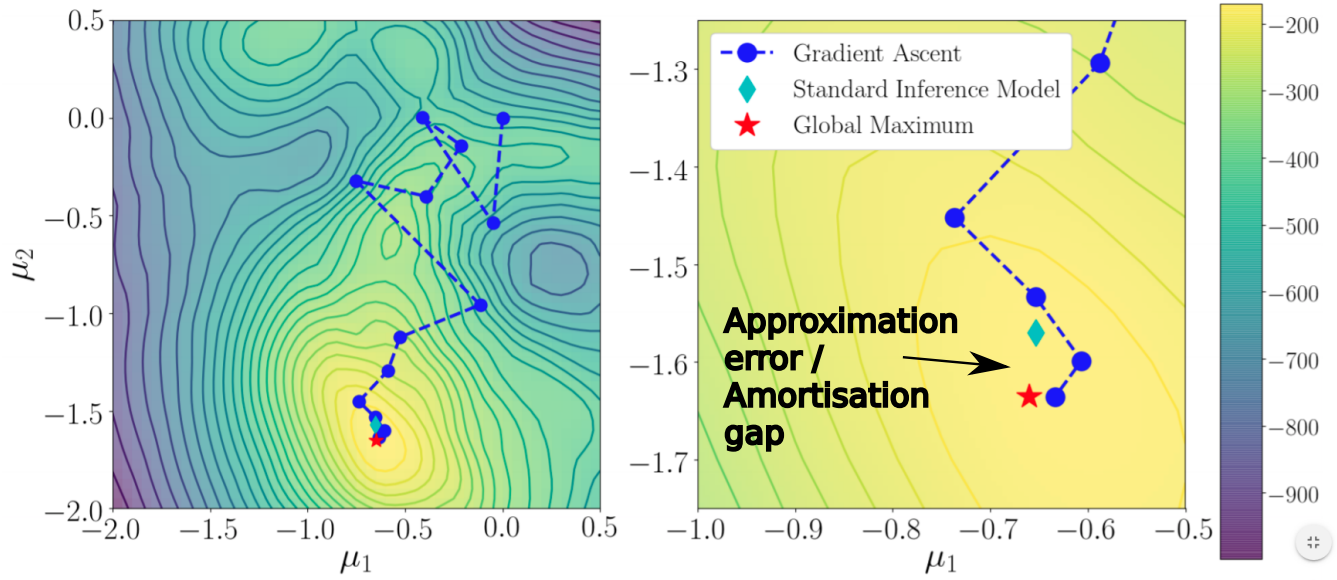



Figure 1 from Marino et al, ICML 2018.

# Issue 2: Maximisation

- The optimisation problem is

$$\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\theta}, \boldsymbol{\phi}}{\operatorname{argmax}} \, \mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, \boldsymbol{\phi}) \tag{23}$$

  where

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \sum_{i=1}^{n} \mathcal{L}_i(\boldsymbol{\theta}, \boldsymbol{\phi}) \tag{24}$$

$$= \sum_{i=1}^{n} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{h}_i|\mathbf{v}_i)} \left[ \log \frac{p(\mathbf{v}_i, \mathbf{h}_i; \boldsymbol{\theta})}{q_{\boldsymbol{\phi}}(\mathbf{h}_i|\mathbf{v}_i)} \right] \tag{25}$$

- We would like to solve it using gradient ascent.
- Difficulties:
  1. We generally cannot compute the expectations in closed form.
  2. The parameter $\boldsymbol{\phi}$ occurs in the expectation so that we cannot pull $\nabla_{\boldsymbol{\phi}}$ inside.

# Reparameterisation

- We can approximate the expectation as a sample average, but we have to keep track of the $\phi$-dependency of the samples.
- For that, let us consider variational distributions $q_\phi(\mathbf{h}|\mathbf{v})$ that can be obtained via a transformation of a random variable $\boldsymbol{\epsilon}$ that we can sample from.

$$\mathbf{h} \sim q_\phi(\mathbf{h}|\mathbf{v}) \qquad \Longleftrightarrow \qquad \mathbf{h} = \mathbf{t}_\phi(\boldsymbol{\epsilon}, \mathbf{v}), \quad \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}) \tag{26}$$

- Examples:
  - $h \sim \mathcal{N}(h; \mu(\mathbf{v}), \sigma^2(\mathbf{v})) \Leftrightarrow h = \mu(\mathbf{v}) + \sigma(\mathbf{v})\epsilon$ with $\epsilon \sim \mathcal{N}(\epsilon, 0, 1)$.
  - Inverse transform sampling
  - Factor analysis or ICA model where factor or mixing matrix depends on $\mathbf{v}$.
  - Normalising flows (not covered in this course)
  - ...

# Reparameterisation

- By the law of the unconscious statistician, we then obtain

$$\mathbb{E}_{q_\phi(\mathbf{h}_i|\mathbf{v}_i)}\left[\log \frac{p(\mathbf{v}_i, \mathbf{h}_i; \boldsymbol{\theta})}{q_\phi(\mathbf{h}_i|\mathbf{v}_i)}\right] = \mathbb{E}_{p(\boldsymbol{\epsilon}_i)}\left[\log \frac{p(\mathbf{v}_i, \mathbf{t}_\phi(\boldsymbol{\epsilon}_i, \mathbf{v}_i); \boldsymbol{\theta})}{q_\phi(\mathbf{t}_\phi(\boldsymbol{\epsilon}_i, \mathbf{v}_i)|\mathbf{v}_i)}\right] \tag{27}$$

- We can now pull the gradients inside

$$\nabla_{\boldsymbol{\theta},\phi}\mathbb{E}_{q_\phi(\mathbf{h}_i|\mathbf{v}_i)}\left[\cdots\right] = \nabla_{\boldsymbol{\theta},\phi}\mathbb{E}_{p(\boldsymbol{\epsilon}_i)}\left[\cdots\right] = \mathbb{E}_{p(\boldsymbol{\epsilon}_i)}\left[\nabla_{\boldsymbol{\theta},\phi}\cdots\right]$$

- The gradient can then be computed via auto-differentiation.
- Note: Alternative to reparameterisation is to use an approach called score function gradient estimation (not examinable).

# Stochastic optimisation

- The gradient of $\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, \boldsymbol{\phi})$ thus becomes

$$\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \sum_{i=1}^{n} \mathbb{E}_{p(\boldsymbol{\epsilon}_i)} \left[ \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \log \frac{p(\mathbf{v}_i, \mathbf{t}_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}_i, \mathbf{v}_i); \boldsymbol{\theta})}{q_{\boldsymbol{\phi}}(\mathbf{t}_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}_i, \mathbf{v}_i) | \mathbf{v}_i)} \right] \quad (28)$$

- We can approximate $\mathbb{E}_{p(\boldsymbol{\epsilon}_i)}$ with a sample average (Monte Carlo integration) with $S$ samples.
- For large $n$ and $S$, evaluation of the gradient is expensive.
- Computing the gradient for all $\mathbf{v}_i$ and using a large $S$ is not necessary. We can use stochastic optimisation instead.
- This means we only evaluate the gradient for a random subset (minibatch) of the $\mathbf{v}_i$ and set $S$ to a small number (e.g. 1!).

We gloss over technical details here; for an introduction to stochastic optimisation, see *Introduction to Stochastic Search and Optimization* by James Spall.

Eq (28) can be manipulated to reduce the variance of the stochastic gradient, see Roeder et al, *Sticking the Landing: Simple, Lower-Variance Gradient Estimators for Variational Inference*, NeuRIPS 2017.

# Program

1. Scalable generic variational learning of latent variable models
   - ELBO for iid data
   - Amortised variational inference
   - Reparameterisation and stochastic optimisation

2. Deep latent variable models and variational autoencoders

# Program

1. Scalable generic variational learning of latent variable models

2. Deep latent variable models and variational autoencoders
   - Deep latent variable model
   - Variational autoencoder (VAE)
   - Gaussian and Bernoulli VAE

# Deep directed graphical models

- Parametric directed graphical models are sets of pdfs/pmfs that factorise as

$$p(\mathbf{x};\boldsymbol{\theta}) = \prod_{k=1}^{d} p(x_k|\mathrm{pa}_k;\boldsymbol{\theta}) \tag{29}$$

where $\mathrm{pa}_k$ denotes the parents of $x_k$ in a given directed acyclic graph (DAG).

- We say that the model is a deep directed graphical model if

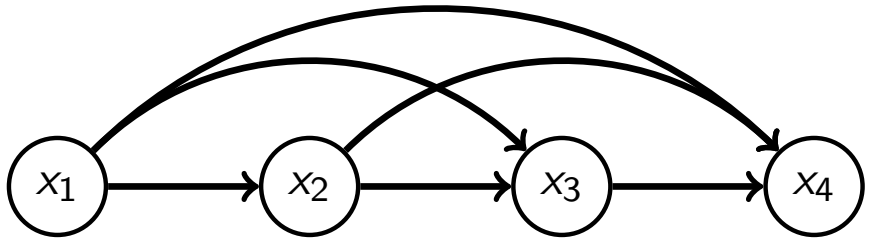
$$p(x_k|\mathrm{pa}_k;\boldsymbol{\theta}) = p(x_k;\boldsymbol{\eta}_k) \quad \text{with} \quad \boldsymbol{\eta}_k = \boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathrm{pa}_k) \tag{30}$$

where $p(x_k;\boldsymbol{\eta})$ is a parametric model and $\boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathrm{pa}_k)$ a parameterised nonlinear function (deep neural network) that maps the parents $\mathrm{pa}_k$ to the model-parameters $\boldsymbol{\eta}_k$.

# Example

- Chain rule $p(\mathbf{x};\boldsymbol{\theta}) = \prod_{k=1}^{d} p(x_k|\mathrm{pre}_k;\boldsymbol{\theta})$ with

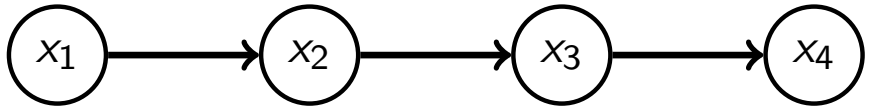
$$p(x_k|\mathrm{pre}_k;\boldsymbol{\theta}) = \mathcal{N}(x_k;\mu_k,\sigma_k^2), \qquad (\mu_k,\sigma_k^2) = \boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathrm{pre}_k)$$




This is one of the autoregressive models from the slides *Basic Assumptions for Efficient Model Representation*.

- Markov chain $p(\mathbf{x};\boldsymbol{\theta}) = \prod_{k=1}^{d} p(x_k|x_{k-1};\boldsymbol{\theta})$ with

$$p(x_k|x_{k-1};\boldsymbol{\theta}) = \mathcal{N}(x_k;\mu_k,\sigma_k^2), \qquad (\mu_k,\sigma_k^2) = \boldsymbol{\eta}_{\boldsymbol{\theta}}^k(x_{k-1})$$



# Deep latent variable model

- A deep (directed) latent variable model is a deep directed graphical model with latent variables.
- Often (but not always), they are models of the form

$$p(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta}) = p(\mathbf{v}|\mathbf{h}; \boldsymbol{\theta})p(\mathbf{h}) \tag{31}$$

where $p(\mathbf{h})$ does not depend on $\boldsymbol{\theta}$ and $p(\mathbf{v}|\mathbf{h}; \boldsymbol{\theta})$ is

$$p(\mathbf{v}|\mathbf{h}; \boldsymbol{\theta}) = \prod_{k=1}^{d} p(v_k|\check{\mathrm{pa}}_k, \mathbf{h}; \boldsymbol{\theta}) \tag{32}$$

with $\check{\mathrm{pa}}_k$ denoting the parents of $v_k$ without $\mathbf{h}$.

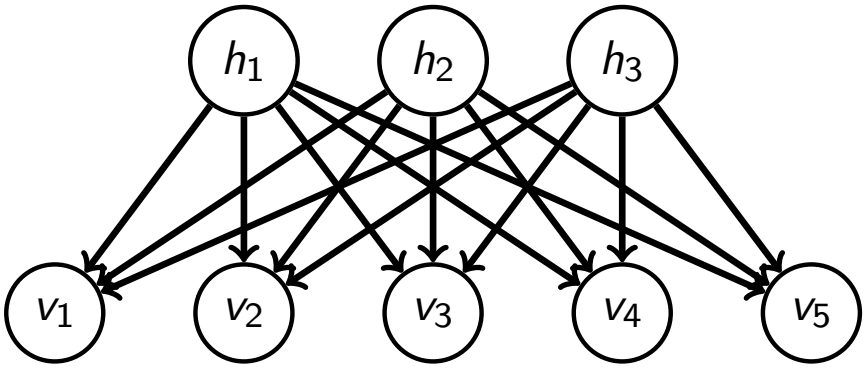
- The conditional is given by

$$p(v_k|\check{\mathrm{pa}}_k, \mathbf{h}; \boldsymbol{\theta}) = p(v_k; \boldsymbol{\eta}_k) \qquad \boldsymbol{\eta}_k = \boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\check{\mathrm{pa}}_k, \mathbf{h}) \tag{33}$$

- Note: parameterised models $p(\mathbf{h}; \boldsymbol{\theta})$ may also be used.

# Graphical model for variational autoencoders

Reconsider the directed acyclic graph for FA and ICA:



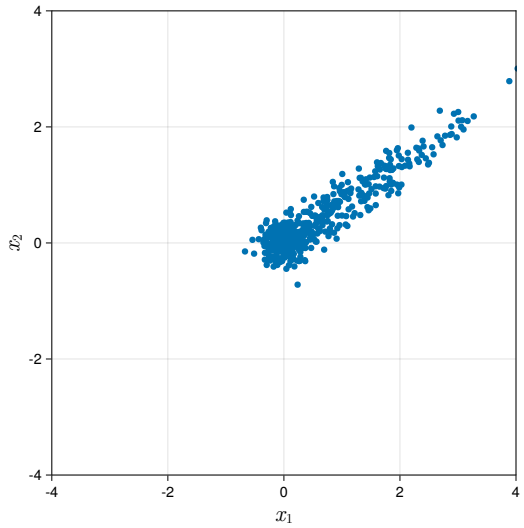

- The visibles $\mathbf{v} = (v_1, \ldots, v_d)$ are independent from each other given the latents $\mathbf{h} = (h_1, \ldots, h_H)$.
- Different assumptions on $p(v_k|\mathbf{h})$ and $p(\mathbf{h})$ give different methods, e.g. FA and ICA.
- Working with $H < d$ and $p(v_k|\mathbf{h}; \boldsymbol{\theta}) = p(v_k; \boldsymbol{\eta}_k)$, where $\boldsymbol{\eta}_k = \boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathbf{h})$, gives variational autoencoders (VAE).
- The function $\boldsymbol{\eta}_k = \boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathbf{h})$ is called the decoder or decoder network.

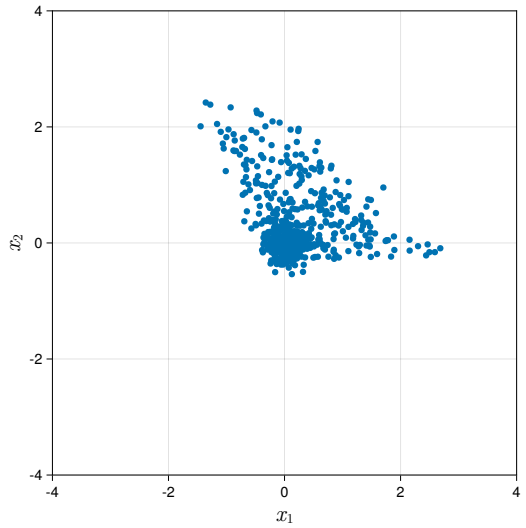
# VAE: overview

- Depending on the data, different parametric families are chosen for the univariate distributions $p(v_k; \boldsymbol{\eta}_k)$
- For example:
  - Gaussian pdf for $v_k \in \mathbb{R}$: Here $\boldsymbol{\eta}_k = (m_k, s_k^2)$ are the mean and variance.
  - Bernoulli pmf for $v_k \in \{0, 1\}$: Here $\boldsymbol{\eta}_k = p_k$ is the probability for $v_k = 1$.
- Note: The parametric families may be simple but the parameter $\boldsymbol{\eta}_k$ is a nonlinear transformation of $\mathbf{h}$, $\boldsymbol{\eta}_k = \boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathbf{h})$, giving rise to a flexible class of models.

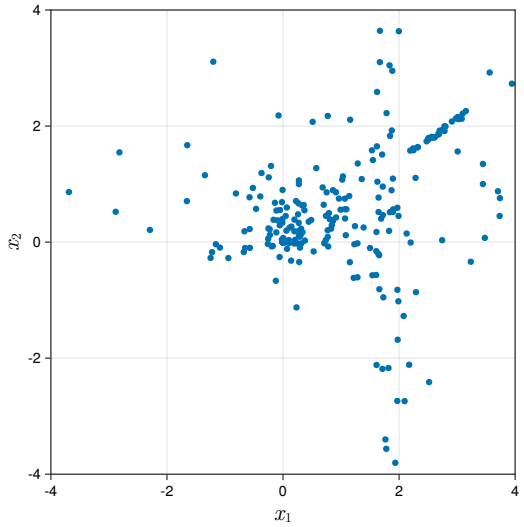
# Example: Gaussian VAE

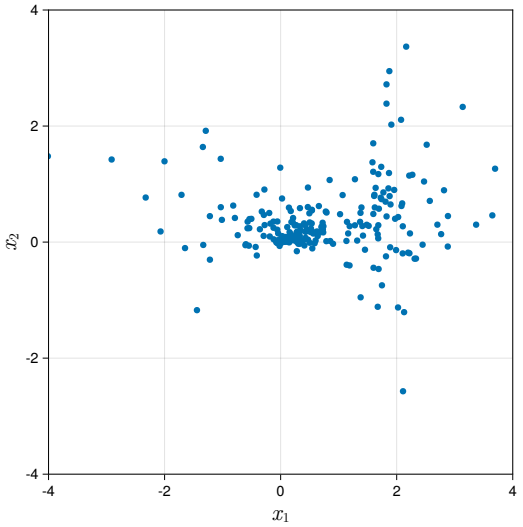
Nonlinear mean function (NN with random weights and ReLu), constant variance:




Nonlinear mean and variance functions:



# VAE: overview

- The variational distribution $q_\phi(\mathbf{h}|\mathbf{v})$ is often assumed to be a factorised Gaussian.
- Variational distribution $q_\phi(\mathbf{h}|\mathbf{v})$ goes under several names: encoder, inference model, or recognition model are used; the model $p(\mathbf{v}|\mathbf{h};\theta)$ is called the decoder or generative model.
- Note: the encoder/decoder names may refer to the distribution or the mapping to their parameters.

# VAE: learning

- We now derive the ELBO for the VAE using that:
  - $p(\mathbf{v}, \mathbf{h}; \theta) = p(\mathbf{v}|\mathbf{h}; \theta)p(\mathbf{h})$ with $p(\mathbf{h}) = \mathcal{N}(\mathbf{h}; \mathbf{0}, \mathbf{I})$
  - Factorised Gaussian for the variational distribution $q_\phi(\mathbf{h}|\mathbf{v})$
- As before:

$$q_\phi(\mathbf{h}|\mathbf{v}) = \prod_k^H q(h_k|\mathbf{v}) \tag{34}$$

$$q_\phi(h_k|\mathbf{v}) = \mathcal{N}(h_k; \mu_k(\mathbf{v}), \sigma_k^2(\mathbf{v})) \tag{35}$$

  That is, $\boldsymbol{\lambda}_\phi(\mathbf{v})$ maps $\mathbf{v}$ to $(\mu_1, \ldots, \mu_H, \sigma_1^2, \ldots, \sigma_H^2)$.
  ($\phi$-dependency of $\mu_k(\mathbf{v}), \sigma_k^2(\mathbf{v})$ is suppressed.)
- With the Gaussianity assumption on $p(\mathbf{h})$ and $q_\phi(\mathbf{h}|\mathbf{v})$, part of the ELBO can be computed in closed form.

# VAE: learning

- We use the following form of the ELBO

$$\mathcal{L}_i = \mathbb{E}_{q_\phi(\mathbf{h}_i|\mathbf{v}_i)}\left[\log p(\mathbf{v}_i|\mathbf{h}_i;\boldsymbol{\theta})\right] - \mathrm{KL}(q_\phi(\mathbf{h}_i|\mathbf{v}_i)||\mathcal{N}(\mathbf{h}_i;\mathbf{0},\mathbf{I}))$$

  First term: reconstruction/fit; second term: regularisation

- The KL-divergence between two Gaussians has a closed-form expression.
- $\mathrm{KL}(q_\phi(\mathbf{h}_i|\mathbf{v}_i)||\mathcal{N}(\mathbf{h}_i;\mathbf{0},\mathbf{I})$ equals

$$\frac{1}{2}\sum_{k=1}^{H}\left(\sigma_k^2(\mathbf{v}_i) + \mu_k^2(\mathbf{v}_i) - 1 - \log(\sigma_k^2(\mathbf{v}_i))\right) \tag{36}$$

- Hence

$$\mathcal{L}_i = \mathbb{E}_{q_\phi(\mathbf{h}_i|\mathbf{v}_i)}\left[\log p(\mathbf{v}_i|\mathbf{h}_i;\boldsymbol{\theta})\right] + \frac{1}{2}\sum_{k=1}^{H}\left(1 + \log(\sigma_k^2(\mathbf{v}_i)) - \sigma_k^2(\mathbf{v}_i) - \mu_k^2(\mathbf{v}_i)\right) \tag{37}$$

# VAE: learning

- With the conditional independence assumption for $p(\mathbf{v}|\mathbf{h};\boldsymbol{\theta})$:

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{h}_i|\mathbf{v}_i)}\left[\log p(\mathbf{v}_i|\mathbf{h}_i;\boldsymbol{\theta})\right] = \sum_{k=1}^{d} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{h}_i|\mathbf{v}_i)}\left[\log p(v_{ik};\boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathbf{h}_i))\right]$$

  where $v_{ik}$ denotes the $k$-th element of $\mathbf{v}_i$.
- We thus have for the VAE:

$$\mathcal{L}_i(\boldsymbol{\theta},\boldsymbol{\phi}) = \sum_{k=1}^{d} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{h}_i|\mathbf{v}_i)}\left[\log p(v_{ik};\boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathbf{h}_i))\right] + \\ + \frac{1}{2}\sum_{k=1}^{H}\left(1 + \log(\sigma_k^2(\mathbf{v}_i)) - \sigma_k^2(\mathbf{v}_i) - \mu_k^2(\mathbf{v}_i)\right) \quad (38)$$

- Optimisation problem

$$\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\theta},\boldsymbol{\phi}}{\operatorname{argmax}}\, \mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta},\boldsymbol{\phi}) = \underset{\boldsymbol{\theta},\boldsymbol{\phi}}{\operatorname{argmax}} \sum_{i=1}^{n} \mathcal{L}_i(\boldsymbol{\theta},\boldsymbol{\phi}) \quad (39)$$

- Solved with stochastic gradient ascent and the reparam. trick.

# Gaussian VAE

- The Gaussian VAE is obtained for

$$p(v_k|\mathbf{h};\boldsymbol{\theta}) = \mathcal{N}(v_k; m_k, s_k^2) \qquad (m_k, s_k^2) = \boldsymbol{\eta}_{\boldsymbol{\theta}}^k(\mathbf{h}) \qquad (40)$$

- Generative model $p(\mathbf{v}|\mathbf{h};\boldsymbol{\theta})$ equivalent to

$$\mathbf{v} = \begin{pmatrix} m_1(\mathbf{h}) \\ \vdots \\ m_D(\mathbf{h}) \end{pmatrix} + \begin{pmatrix} s_1(\mathbf{h}) & & \\ & \ddots & \\ & & s_D(\mathbf{h}) \end{pmatrix} \mathbf{n}, \quad \mathbf{n} \sim \mathcal{N}(\mathbf{n}; \mathbf{0}, \mathbf{I})$$

- FA obtained for $\mathbf{m} = (m_1, \ldots, m_D)^\top = \mathbf{F}\mathbf{h} + \mathbf{c}$ and $s_k^2 = \Psi_k$.
- Gaussian VAE is a nonlinear generalisation of FA.

# Bernoulli VAE

- The Bernoulli VAE with $v_k \in \{0, 1\}$ is obtained for

$$p(v_k|\mathbf{h}; \boldsymbol{\theta}) = p_k^{v_k}(1 - p_k)^{(1-v_k)} \qquad p_k = \eta_\theta^k(\mathbf{h}) \tag{41}$$

- This is often also used for $v_k \in [0, 1]$. While the ELBO can be evaluated, it is formally wrong since $v_k$ is not binary.
- For $v_k \in [0, 1]$, use the so-called continuous Bernoulli distribution or the beta distribution instead.

  (see Loaiza-Ganem and Cunningham, *The continuous Bernoulli: fixing a pervasive error in variational autoencoders*, NeuRIPS 2019)

# Program recap

1. Scalable generic variational learning of latent variable models
   - ELBO for iid data
   - Amortised variational inference
   - Reparameterisation and stochastic optimisation

2. Deep latent variable models and variational autoencoders
   - Deep latent variable model
   - Variational autoencoder (VAE)
   - Gaussian and Bernoulli VAE