Probabilistic & Unsupervised Learning
Approximate Inference

Parametric Variational Methods
and Recognition Models

Maneesh Sahani
maneesh@gatsby.ucl.ac.uk

Gatsby Computational Neuroscience Unit, and
MSc ML/CSML, Dept Computer Science
University College London

Term 1, Autumn 2024

Variational methods

- Our treatment of variational methods has (except EP) emphasised ‘natural’ choices of variational family – often factorised using the same functional (ExpFam) form as joint.
  - mostly restricted to joint exponential families – facilitates hierarchical and distributed models, but not non-linear/non-conjugate.

Variational methods

- Our treatment of variational methods has (except EP) emphasised 'natural' choices of variational family – often factorised using the same functional (ExpFam) form as joint.
  - mostly restricted to joint exponential families – facilitates hierarchical and distributed models, but not non-linear/non-conjugate.

- Consider parametric variational approximations using a constrained family $q(\mathcal{Z};\rho)$.

  The constrained (approximate) variational E-step becomes:

$$q(\mathcal{Z}) := \underset{q\in\{q(\mathcal{Z};\rho)\}}{\operatorname{argmax}} \mathcal{F}\big(q(\mathcal{Z}),\theta^{(k-1)}\big) \quad \Rightarrow \quad \rho^{(k)} := \underset{\rho}{\operatorname{argmax}} \mathcal{F}\big(q(\mathcal{Z};\rho),\theta^{(k-1)}\big)$$

  and so we can replace constrained optimisation of $\mathcal{F}(q,\theta)$ with unconstrained optimisation of a constrained $\mathcal{F}(\rho,\theta)$:

$$\mathcal{F}(\rho,\theta) = \left\langle \log P(\mathcal{X},\mathcal{Z}|\theta^{(k-1)}) \right\rangle_{q(\mathcal{Z};\rho)} + \mathbf{H}[\rho]$$

  It might still be valuable to use coordinate ascent in ρ and θ, although this is no longer necessary.

Optimising the variational parameters

$$\mathcal{F}(\rho, \theta) = \left\langle \log P(\mathcal{X}, \mathcal{Z}|\theta^{(k-1)}) \right\rangle_{q(\mathcal{Z};\rho)} + \mathbf{H}[\rho]$$

- In some special cases, the expectations of the log-joint under $q(\mathcal{Z}; \rho)$ can be expressed in closed form, but these are rare.

Optimising the variational parameters

$$\mathcal{F}(\rho, \theta) = \left\langle \log P(\mathcal{X}, \mathcal{Z} | \theta^{(k-1)}) \right\rangle_{q(\mathcal{Z};\rho)} + \mathbf{H}[\rho]$$

- In some special cases, the expectations of the log-joint under $q(\mathcal{Z}; \rho)$ can be expressed in closed form, but these are rare.
- Otherwise we might seek to follow $\nabla_\rho \mathcal{F}$.

Optimising the variational parameters

$$\mathcal{F}(\rho, \theta) = \left\langle \log P(\mathcal{X}, \mathcal{Z} | \theta^{(k-1)}) \right\rangle_{q(\mathcal{Z};\rho)} + \mathbf{H}[\rho]$$

- In some special cases, the expectations of the log-joint under $q(\mathcal{Z}; \rho)$ can be expressed in closed form, but these are rare.
- Otherwise we might seek to follow $\nabla_\rho \mathcal{F}$.
- Naively, this requires evaluting a high-dimensional expectation wrt $q(\mathcal{Z}, \rho)$ as a function of ρ – not simple.

Optimising the variational parameters

$$\mathcal{F}(\rho, \theta) = \left\langle \log P(\mathcal{X}, \mathcal{Z} | \theta^{(k-1)}) \right\rangle_{q(\mathcal{Z};\rho)} + \mathbf{H}[\rho]$$

- In some special cases, the expectations of the log-joint under $q(\mathcal{Z}; \rho)$ can be expressed in closed form, but these are rare.
- Otherwise we might seek to follow $\nabla_\rho \mathcal{F}$.
- Naively, this requires evaluting a high-dimensional expectation wrt $q(\mathcal{Z}, \rho)$ as a function of ρ – not simple.
- At least three solutions:

Optimising the variational parameters

$$\mathcal{F}(\rho, \theta) = \left\langle \log P(\mathcal{X}, \mathcal{Z} | \theta^{(k-1)}) \right\rangle_{q(\mathcal{Z};\rho)} + \mathbf{H}[\rho]$$

- In some special cases, the expectations of the log-joint under $q(\mathcal{Z}; \rho)$ can be expressed in closed form, but these are rare.
- Otherwise we might seek to follow $\nabla_\rho \mathcal{F}$.
- Naively, this requires evaluting a high-dimensional expectation wrt $q(\mathcal{Z}, \rho)$ as a function of ρ – not simple.
- At least three solutions:
  - "Score-based" gradient estimate, and Monte-Carlo (Ranganath et al. 2014).

Optimising the variational parameters

$$\mathcal{F}(\rho, \theta) = \left\langle \log P(\mathcal{X}, \mathcal{Z}|\theta^{(k-1)}) \right\rangle_{q(\mathcal{Z};\rho)} + \mathbf{H}[\rho]$$

- In some special cases, the expectations of the log-joint under $q(\mathcal{Z}; \rho)$ can be expressed in closed form, but these are rare.
- Otherwise we might seek to follow $\nabla_\rho \mathcal{F}$.
- Naively, this requires evaluting a high-dimensional expectation wrt $q(\mathcal{Z}, \rho)$ as a function of ρ – not simple.
- At least three solutions:
  - "Score-based" gradient estimate, and Monte-Carlo (Ranganath et al. 2014).
  - Recognition network trained in separate phase – not strictly variational (Dayan et al. 1995).

Optimising the variational parameters

$$\mathcal{F}(\rho, \theta) = \left\langle \log P(\mathcal{X}, \mathcal{Z}|\theta^{(k-1)}) \right\rangle_{q(\mathcal{Z};\rho)} + \mathbf{H}[\rho]$$

- In some special cases, the expectations of the log-joint under $q(\mathcal{Z}; \rho)$ can be expressed in closed form, but these are rare.
- Otherwise we might seek to follow $\nabla_\rho \mathcal{F}$.
- Naively, this requires evaluting a high-dimensional expectation wrt $q(\mathcal{Z}, \rho)$ as a function of ρ – not simple.
- At least three solutions:
  - "Score-based" gradient estimate, and Monte-Carlo (Ranganath et al. 2014).
  - Recognition network trained in separate phase – not strictly variational (Dayan et al. 1995).
  - Recognition network trained simultaneously with generative model using "frozen" samples (Kingma and Welling 2014; Rezende et al. 2014).

Score-based gradient estimate

We have:

$$
\begin{aligned}
\nabla_\rho \mathcal{F}(\rho, \theta) &= \nabla_\rho \int d\mathcal{Z}\, q(\mathcal{Z}; \rho)(\log P(\mathcal{X}, \mathcal{Z}|\theta) - \log q(\mathcal{Z}; \rho)) \\
&= \int d\mathcal{Z} \Big([\nabla_\rho q(\mathcal{Z}; \rho)](\log P(\mathcal{X}, \mathcal{Z}|\theta) - \log q(\mathcal{Z}; \rho)) \\
&\qquad + q(\mathcal{Z}; \rho)\nabla_\rho[\log P(\mathcal{X}, \mathcal{Z}|\theta) - \log q(\mathcal{Z}; \rho)] \Big)
\end{aligned}
$$

Score-based gradient estimate

We have:

$$\nabla_\rho \mathcal{F}(\rho,\theta) = \nabla_\rho \int d\mathcal{Z}\, q(\mathcal{Z};\rho)(\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho))$$

$$= \int d\mathcal{Z} \Big([\nabla_\rho q(\mathcal{Z};\rho)](\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho))$$

$$+ q(\mathcal{Z};\rho)\nabla_\rho[\cancel{\log P(\mathcal{X},\mathcal{Z}|\theta)} - \log q(\mathcal{Z};\rho)] \Big)$$

Now,

$$\nabla_\rho \log P(\mathcal{X},\mathcal{Z}|\theta) = 0 \qquad \text{(no direct dependence)}$$

Score-based gradient estimate

We have:

$$\nabla_\rho \mathcal{F}(\rho,\theta) = \nabla_\rho \int d\mathcal{Z}\, q(\mathcal{Z};\rho)(\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho))$$

$$= \int d\mathcal{Z} \Big([\nabla_\rho q(\mathcal{Z};\rho)](\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho))$$

$$+ q(\mathcal{Z};\rho)\nabla_\rho[\cancel{\log P(\mathcal{X},\mathcal{Z}|\theta)} - \cancel{\log q(\mathcal{Z};\rho)}] \Big)$$

Now,

$$\nabla_\rho \log P(\mathcal{X},\mathcal{Z}|\theta) = 0 \qquad \text{(no direct dependence)}$$

$$\int d\mathcal{Z}\, q(\mathcal{Z};\rho)\nabla_\rho \log q(\mathcal{Z};\rho) = \int d\mathcal{Z}\, \nabla_\rho q(\mathcal{Z};\rho) = 0 \qquad \text{(always normalised)}$$

Score-based gradient estimate

We have:

$$\begin{aligned}
\nabla_\rho \mathcal{F}(\rho,\theta) &= \nabla_\rho \int d\mathcal{Z}\, q(\mathcal{Z};\rho)(\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho)) \\
&= \int d\mathcal{Z} \Big([\nabla_\rho q(\mathcal{Z};\rho)](\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho)) \\
&\qquad + q(\mathcal{Z};\rho)\nabla_\rho[\cancel{\log P(\mathcal{X},\mathcal{Z}|\theta)} - \cancel{\log q(\mathcal{Z};\rho)}] \Big)
\end{aligned}$$

Now,

$$\nabla_\rho \log P(\mathcal{X},\mathcal{Z}|\theta) = 0 \qquad \text{(no direct dependence)}$$

$$\int d\mathcal{Z}\, q(\mathcal{Z};\rho)\nabla_\rho \log q(\mathcal{Z};\rho) = \int d\mathcal{Z}\, \nabla_\rho q(\mathcal{Z};\rho) = 0 \qquad \text{(always normalised)}$$

$$\nabla_\rho q(\mathcal{Z};\rho) = q(\mathcal{Z};\rho)\nabla_\rho \log q(\mathcal{Z};\rho) \qquad \leftarrow \text{“score trick”}$$

Score-based gradient estimate

We have:

$$\nabla_\rho \mathcal{F}(\rho,\theta) = \nabla_\rho \int d\mathcal{Z}\, q(\mathcal{Z};\rho)(\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho))$$

$$= \int d\mathcal{Z} \Big([\nabla_\rho q(\mathcal{Z};\rho)](\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho))$$

$$+ q(\mathcal{Z};\rho)\nabla_\rho[\cancel{\log P(\mathcal{X},\mathcal{Z}|\theta)} - \cancel{\log q(\mathcal{Z};\rho)}] \Big)$$

Now,

$$\nabla_\rho \log P(\mathcal{X},\mathcal{Z}|\theta) = 0 \qquad \text{(no direct dependence)}$$

$$\int d\mathcal{Z}\, q(\mathcal{Z};\rho)\nabla_\rho \log q(\mathcal{Z};\rho) = \int d\mathcal{Z}\, \nabla_\rho q(\mathcal{Z};\rho) = 0 \qquad \text{(always normalised)}$$

$$\nabla_\rho q(\mathcal{Z};\rho) = q(\mathcal{Z};\rho)\nabla_\rho \log q(\mathcal{Z};\rho) \qquad \leftarrow \text{"score trick"}$$

So,

$$\nabla_\rho \mathcal{F}(\rho,\theta) = \Big\langle [\nabla_\rho \log q(\mathcal{Z};\rho)](\log P(\mathcal{X},\mathcal{Z}|\theta) - \log q(\mathcal{Z};\rho)) \Big\rangle_{q(\mathcal{Z};\rho)}$$

Score-based gradient estimate

We have:

$$\begin{aligned}
\nabla_\rho \mathcal{F}(\rho, \theta) &= \nabla_\rho \int d\mathcal{Z}\, q(\mathcal{Z}; \rho)(\log P(\mathcal{X}, \mathcal{Z}|\theta) - \log q(\mathcal{Z}; \rho)) \\
&= \int d\mathcal{Z} \Big([\nabla_\rho q(\mathcal{Z}; \rho)](\log P(\mathcal{X}, \mathcal{Z}|\theta) - \log q(\mathcal{Z}; \rho)) \\
&\qquad\qquad + q(\mathcal{Z}; \rho) \nabla_\rho [\cancel{\log P(\mathcal{X}, \mathcal{Z}|\theta)} - \cancel{\log q(\mathcal{Z}; \rho)}] \Big)
\end{aligned}$$

Now,

$$\nabla_\rho \log P(\mathcal{X}, \mathcal{Z}|\theta) = 0 \qquad \text{(no direct dependence)}$$

$$\int d\mathcal{Z}\, q(\mathcal{Z}; \rho) \nabla_\rho \log q(\mathcal{Z}; \rho) = \int d\mathcal{Z}\, \nabla_\rho q(\mathcal{Z}; \rho) = 0 \qquad \text{(always normalised)}$$

$$\nabla_\rho q(\mathcal{Z}; \rho) = q(\mathcal{Z}; \rho) \nabla_\rho \log q(\mathcal{Z}; \rho) \qquad \leftarrow \text{“score trick”}$$

So,

$$\nabla_\rho \mathcal{F}(\rho, \theta) = \Big\langle [\nabla_\rho \log q(\mathcal{Z}; \rho)](\log P(\mathcal{X}, \mathcal{Z}|\theta) - \log q(\mathcal{Z}; \rho)) \Big\rangle_{q(\mathcal{Z}; \rho)}$$

Reduced gradient of expectation to expectation of gradient – easier to compute. Also called the REINFORCE trick.

Factorisation

$$\nabla_\rho \mathcal{F}(\rho, \theta) = \Big\langle [\nabla_\rho \log q(\mathcal{Z}; \rho)](\log P(\mathcal{X}, \mathcal{Z}|\theta) - \log q(\mathcal{Z}; \rho)) \Big\rangle_{q(\mathcal{Z};\rho)}$$

- Still requires a high-dimensional expectation, but can now be evaluated by Monte-Carlo.
- Dimensionality reduced by factorisation (particularly where $P(\mathcal{X}, \mathcal{Z})$ is factorised).

Let $q(\mathcal{Z}) = \prod_i q(\mathcal{Z}_i|\rho_i)$ factor over disjoint cliques; let $\bar{\mathcal{Z}}_i$ be the minimal Markov blanket of $\mathcal{Z}_i$ in the joint; $P_{\bar{\mathcal{Z}}_i}$ be the product of joint factors that include any element of $\mathcal{Z}_i$ (so the union of their arguments is $\bar{\mathcal{Z}}_i$); and $P_{\neg\bar{\mathcal{Z}}_i}$ the remaining factors. Then,

$$\begin{aligned}
\nabla_{\rho_i} \mathcal{F}(\{\rho_j\}, \theta) &= \Big\langle [\nabla_{\rho_i} \textstyle\sum_j \log q(\mathcal{Z}_j; \rho_j)](\log P(\mathcal{X}, \mathcal{Z}|\theta) - \textstyle\sum_j \log q(\mathcal{Z}_j; \rho_j)) \Big\rangle_{q(\mathcal{Z})} \\
&= \Big\langle [\nabla_{\rho_i} \log q(\mathcal{Z}_i; \rho_i)](\log P_{\bar{\mathcal{Z}}_i}(\mathcal{X}, \bar{\mathcal{Z}}_i) - \log q(\mathcal{Z}_i; \rho_i)) \Big\rangle_{q(\bar{\mathcal{Z}}_i)} \\
&\quad + \Big\langle [\nabla_{\rho_i} \log q(\mathcal{Z}_i; \rho_i)] \underbrace{(\log P_{\neg\bar{\mathcal{Z}}_i}(\mathcal{X}, \mathcal{Z}_{\neg i}) - \sum_{j \neq i} \log q(\mathcal{Z}_j; \rho_j)}_{\text{constant wrt } \mathcal{Z}_i} \Big\rangle_{q(\mathcal{Z})}
\end{aligned}$$

So the second term is proportional to $\langle \nabla_{\rho_i} \log q(\mathcal{Z}_i; \rho_i) \rangle_{q(\mathcal{Z}_i)}$, this $= 0$ as before.

So expectations are only needed wrt $q(\bar{\mathcal{Z}}_i)$ $\rightarrow$ variational message passing!

Sampling

So the “black-box” variational approach is as follows:

- Choose a parametric (factored) variational family $q(\mathcal{Z}) = \prod_i q(\mathcal{Z}_i; \rho_i)$.
- Initialise factors.
- Repeat to convergence:
  - **Stochastic VE-step**. For each *i*:
    - Sample from $q(\bar{\mathcal{Z}}_i)$ and estimate expected gradient $\nabla_{\rho_i}\mathcal{F}$.
    - Update ρ_i along gradient.
  - **Stochastic M-step**. For each *i*:
    - Sample from each $q(\bar{\mathcal{Z}}_i)$.
    - Update corresponding parameters.

- Stochastic gradient steps may employ a Robbins-Munro step-size sequence to promote convergence.
- Variance of the gradient estimators can also be controlled by clever Monte-Carlo techniques (orginal authors used a “control variate” method that we have not studied).

Recognition Models

We have not generally distinguished between multivariate models and iid data instances, grouping all variables together in $\mathcal{Z}$.

However, even for large models (such as HMMs), we often work with multiple data draws (e.g. multiple strings) and each instance requires a separate variational optimisation.

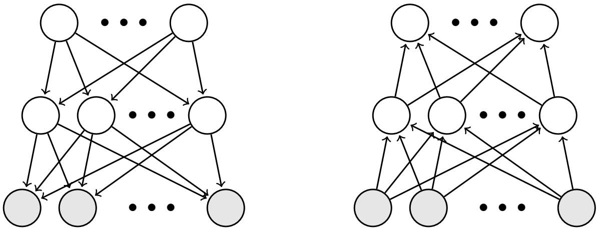
Suppose that we have fixed length vectors $\{(\mathbf{x}_i, \mathbf{z}_i)\}$ ($\mathbf{z}$ is still latent).

- Optimal variational distribution $q^*(\mathbf{z}_i)$ depends on $\mathbf{x}_i$.
- Learn this mapping (in parametric form): $q\big(\mathbf{z}_i; \rho = f(\mathbf{x}_i; \phi)\big)$.
- Now ρ is the output of a general function approximator f (a GP, neural network or similar) parametrised by ϕ, trained to map $\mathbf{x}_i$ to the variational parameters of $q(\mathbf{z}_i)$.
- The mapping function f is called a recognition model.
- This is approach is now often called amortised inference.

How to learn f?

The Helmholtz Machine

Dayan et al. (1995) originally studied binary sigmoid belief net, with parallel recognition model:



Two phase learning:

- Wake phase: given current f, estimate mean-field representation from data (mean sufficient stats for Bernoulli are just probabilities):

  $$q(\mathbf{z}_i) = \text{Bernoulli}[\hat{\mathbf{z}}_i] \qquad \hat{\mathbf{z}}_i = f(\mathbf{x}_i; \phi)$$

  Update generative parameters θ according to $\nabla_\theta \mathcal{F}(\{\hat{\mathbf{z}}_i\}, \theta)$.

- Sleep phase: sample $\{\mathbf{z}_s, \mathbf{x}_s\}_{s=1}^S$ from current generative model. Update recognition parameters ϕ to direct $f(\mathbf{x}_s)$ towards $\mathbf{z}_s$ (simple gradient learning).

  $$\Delta\phi \propto \sum_s (\mathbf{z}_s - f(\mathbf{x}_s; \phi)) \nabla_\phi f(\mathbf{x}_s; \phi)$$

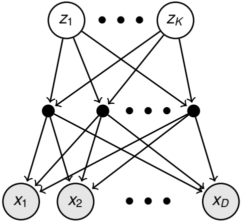
The Helmholtz Machine

- Can sample $\mathbf{z}$ from recognition model rather than just evaluate means.
  - Expectations in free-energy can be computed directly rather than by mean substitution.
  - In hierarchical models, output of higher recognition layers then depends on samples at previous stages, which introduces correlations between samples at different layers.
- Recognition model structure need not exactly echo generative model.
- More general approach is to train f to yield mean parameters of ExpFam $q(\mathbf{z})$ (later).
- Sleep phase learning minimises $\mathbf{KL}[p_\theta(\mathbf{z}|\mathbf{x})\|q(\mathbf{z}; f(\mathbf{x}, \phi))]$. Opposite to variational objective, but may not matter if divergence is small enough.

Variational Autoencoders

- Fuse wake and sleep phases, optimising $\mathcal{F}$ wrt generative and recognition parameters using reparametrisation.

Variational Autoencoders

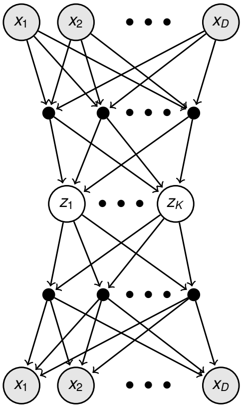


- Fuse wake and sleep phases, optimising $\mathcal{F}$ wrt generative and recognition parameters using reparametrisation.
- Canonical generative conditional is Gaussian with NN (usually MLP) mean (variance may also be parametrised by another NN):

$$\mathrm{P}(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathrm{I})$$

$$\mathrm{P}(\mathbf{x}|\mathbf{z}) = \mathcal{N}\left(\mathbf{g}_{\mathrm{NN}}(\mathbf{z}; \boldsymbol{\theta}), \sigma^2 \mathrm{I}\right)$$

Variational Autoencoders

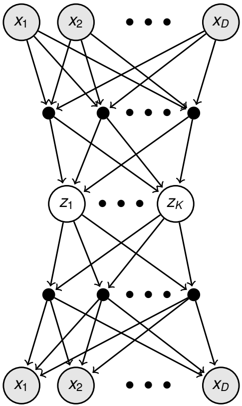


- Fuse wake and sleep phases, optimising $\mathcal{F}$ wrt generative and recognition parameters using reparametrisation.
- Canonical generative conditional is Gaussian with NN (usually MLP) mean (variance may also be parametrised by another NN):

$$\mathrm{P}(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathrm{I})$$
$$\mathrm{P}(\mathbf{x}|\mathbf{z}) = \mathcal{N}\left(\mathbf{g}_{\mathrm{NN}}(\mathbf{z}; \boldsymbol{\theta}), \sigma^2 \mathrm{I}\right)$$

- NN recognition model estimates parameters of posterior: $q(\mathbf{z}|\mathbf{x}; \mathbf{f}(\mathbf{x}, \phi))$.

Variational Autoencoders



- Fuse wake and sleep phases, optimising $\mathcal{F}$ wrt generative and recognition parameters using reparametrisation.
- Canonical generative conditional is Gaussian with NN (usually MLP) mean (variance may also be parametrised by another NN):

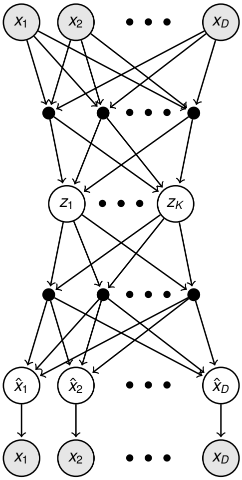
$$\mathrm{P}(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathrm{I})$$
$$\mathrm{P}(\mathbf{x}|\mathbf{z}) = \mathcal{N}\left(\mathbf{g}_{\mathrm{NN}}(\mathbf{z}; \boldsymbol{\theta}), \sigma^2 \mathrm{I}\right)$$

- NN recognition model estimates parameters of posterior: $q(\mathbf{z}|\mathbf{x}; \mathbf{f}(\mathbf{x}, \phi))$.
- Free energy:

$$\mathcal{F}(\theta, \phi) = \sum_{\text{data}} \langle \log \mathrm{P}(\mathbf{x}|\mathbf{z}) \rangle_{q(\mathbf{z}|\mathbf{x})} - \mathbf{KL}[q(\mathbf{z}|\mathbf{x}) \| \mathrm{P}(\mathbf{z})]$$

$$= -\sum_{\text{data}} \left\langle \frac{\|\mathbf{x} - \mathbf{g}(\mathbf{z})\|^2}{2\sigma^2} \right\rangle_q + \mathbf{KL}[q(\mathbf{z}|\mathbf{x}) \| \mathrm{P}(\mathbf{z})]$$

Variational Autoencoders



- Fuse wake and sleep phases, optimising $\mathcal{F}$ wrt generative and recognition parameters using reparametrisation.
- Canonical generative conditional is Gaussian with NN (usually MLP) mean (variance may also be parametrised by another NN):

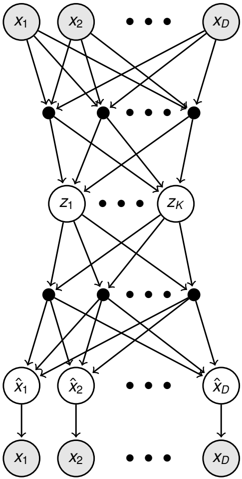
$$\mathrm{P}(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathrm{I})$$
$$\mathrm{P}(\mathbf{x}|\mathbf{z}) = \mathcal{N}\left(\mathbf{g}_{\mathrm{NN}}(\mathbf{z}; \boldsymbol{\theta}), \sigma^2 \mathrm{I}\right)$$

- NN recognition model estimates parameters of posterior: $q(\mathbf{z}|\mathbf{x}; \mathbf{f}(\mathbf{x}, \phi))$.
- Free energy:

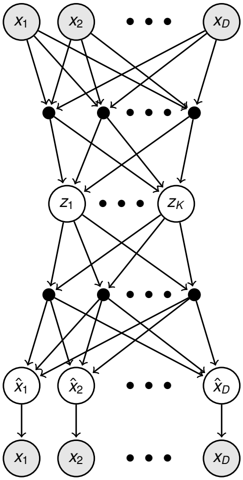
$$\mathcal{F}(\theta, \phi) = \sum_{\text{data}} \langle \log \mathrm{P}(\mathbf{x}|\mathbf{z}) \rangle_{q(\mathbf{z}|\mathbf{x})} - \mathbf{KL}[q(\mathbf{z}|\mathbf{x}) \| \mathrm{P}(\mathbf{z})]$$

$$= -\sum_{\text{data}} \underbrace{\left\langle \frac{\|\mathbf{x} - \overbrace{\mathbf{g}(\mathbf{z})}^{\hat{\mathbf{x}} \;\leftarrow\; \text{“reconstruction”}}\|^2}{2\sigma^2} \right\rangle_q}_{\text{“reconstruction cost”}} + \underbrace{\mathbf{KL}[q(\mathbf{z}|\mathbf{x}) \| \mathrm{P}(\mathbf{z})]}_{\text{“regulariser”}}$$

Variational Autoencoders

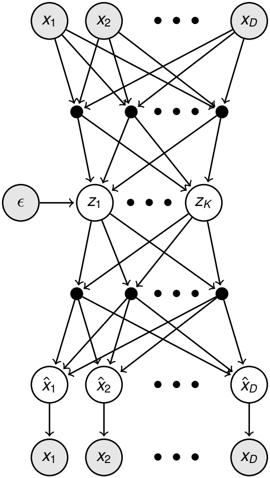


Variational Autoencoders



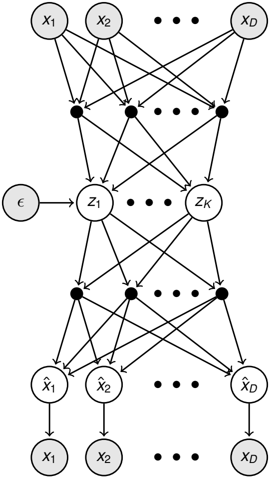
- The expectation of a non-linear (NN) function is intractable.

Variational Autoencoders



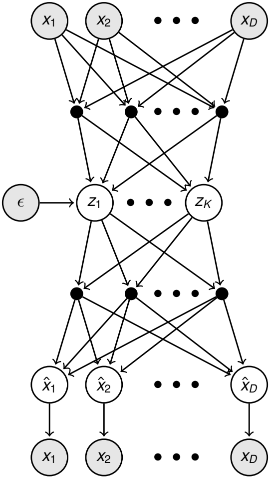
- The expectation of a non-linear (NN) function is intractable.
- Generate S samples from $q(\mathbf{z}|\mathbf{x})$ using deterministic transformation of standard random variates (reparametrisation trick).

Variational Autoencoders



- The expectation of a non-linear (NN) function is intractable.
- Generate S samples from $q(\mathbf{z}|\mathbf{x})$ using deterministic transformation of standard random variates (reparametrisation trick).
  - E.g. if $\mathbf{f}$ gives marginal μ_i and σ_i for latents z_i and $\epsilon_i^s \sim \mathcal{N}(0, 1)$, then $z_i^s = \mu_i + \sigma_i \epsilon_i^s$.

Variational Autoencoders



- The expectation of a non-linear (NN) function is intractable.
- Generate S samples from $q(\mathbf{z}|\mathbf{x})$ using deterministic transformation of standard random variates (reparametrisation trick).
  - E.g. if $\mathbf{f}$ gives marginal μ_i and σ_i for latents z_i and $\epsilon_i^s \sim \mathcal{N}(0,1)$, then $z_i^s = \mu_i + \sigma_i \epsilon_i^s$.
- Now generative and recognition parameters can be trained together by gradient descent (backprop), holding ϵ^s fixed.

$$\mathcal{F}_i(\theta, \phi) = \frac{1}{S} \sum_s \log P(\mathbf{x}_i, \mathbf{z}_i^s; \theta) - \log q(\mathbf{z}_i^s; \mathbf{f}(\mathbf{x}_i, \phi))$$

$$\frac{\partial}{\partial \theta} \mathcal{F}_i = \frac{1}{S} \sum_s \nabla_\theta \log P(\mathbf{x}_i, \mathbf{z}_i^s; \theta)$$

$$\frac{\partial}{\partial \phi} \mathcal{F}_i = \frac{1}{S} \sum_s \frac{\partial}{\partial \mathbf{z}_i^s} (\log P(\mathbf{x}_i, \mathbf{z}_i^s; \theta) - \log q(\mathbf{z}_i^s; \mathbf{f}(\mathbf{x}_i))) \frac{d\mathbf{z}_i^s}{d\phi}$$

$$+ \frac{\partial}{\partial \mathbf{f}(\mathbf{x}_i)} \log q(\mathbf{z}_i^s; \mathbf{f}(\mathbf{x}_i)) \frac{d\mathbf{f}(\mathbf{x}_i)}{d\phi}$$

Variational Autoencoders

- Frozen samples ϵ^s can be redrawn to avoid overfitting.
- May be possible to evaluate entropy and $\langle \log P(\mathbf{z}) \rangle$ without sampling, reducing variance.
- Differentiable reparametrisations are available for a number of different distributions.
  - requires approximation for discrete-valued variables (Gumbel or "concrete" distributions)
- Conditional $P(\mathbf{x}|\mathbf{z}, \theta)$ may be more complex: RNNs, transformers, ...
  - May include internal stochastic nodes: requires recognition network to estimate all distributions (see "ladder VAE").
  - In practice, hierarchical models appear difficult to learn.

More recent work

- Changing the variational cost function (tightening the bound):
  - Importance-Weighted autoencoder (IWAE)
  - Filtering variational objective (FIVO)
  - Thermodynamic variational objective (TVO)
- Flexible variational distributions (and avoiding inference)
  - Normalising flows
  - DDC-Helmholtz machine
  - Amortised learning
  - Diffusion models
- Structured generative models
  - "standard" VAE generative model both too powerful and too simple for learning
  - local conjugate inference – structured VAEs
- Recognition-parametrised models
  - RPMs model (latent-induced) joint dependence, but not marginals of observations

Far from exhaustive . . . these are all areas of active research. We'll survey a few ideas.

Importance-weighted free energy

Another interpretation of $\mathcal{F}$: Jensen bound on importance sampled estimate.

$$\ell(\theta) = \log \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z}|\mathbf{x})}[p(\mathbf{x})]$$

$\Leftrightarrow$ marginalising $\mathbf{z}$ from joint

Importance-weighted free energy

Another interpretation of $\mathcal{F}$: Jensen bound on importance sampled estimate.

$$\ell(\theta) = \log \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}[p(\mathbf{x})] = \log \mathbb{E}_{\mathbf{z}\sim q}\left[\frac{p(\mathbf{x})p(\mathbf{z}|\mathbf{x})}{q(\mathbf{z})}\right]$$

Importance-weighted free energy

Another interpretation of $\mathcal{F}$: Jensen bound on importance sampled estimate.

$$\ell(\theta) = \log \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z}|\mathbf{x})}[p(\mathbf{x})] = \log \mathbb{E}_{\mathbf{z} \sim q}\left[\frac{p(\mathbf{x})p(\mathbf{z}|\mathbf{x})}{q(\mathbf{z})}\right] \geq \mathbb{E}_{\mathbf{z} \sim q}\left[\log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z})}\right]$$

Importance-weighted free energy

Another interpretation of $\mathcal{F}$: Jensen bound on importance sampled estimate.

$$\ell(\theta) = \log \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}[p(\mathbf{x})] = \log \mathbb{E}_{\mathbf{z}\sim q}\left[\frac{p(\mathbf{x})p(\mathbf{z}|\mathbf{x})}{q(\mathbf{z})}\right] \geq \mathbb{E}_{\mathbf{z}\sim q}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}\right]$$

So

proposal

$$\mathcal{F}(q,\theta) = \left\langle \log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z})} \right\rangle_q = \mathbb{E}_{\mathbf{z}\sim q}\left[\log p(\mathbf{x}) \frac{p(\mathbf{z}|\mathbf{x})}{q(\mathbf{z})}\right]$$

-

importance weight

Importance-weighted free energy

Another interpretation of $\mathcal{F}$: Jensen bound on importance sampled estimate.

$$\ell(\theta) = \log \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}[p(\mathbf{x})] = \log \mathbb{E}_{\mathbf{z}\sim q}\left[\frac{p(\mathbf{x})p(\mathbf{z}|\mathbf{x})}{q(\mathbf{z})}\right] \geq \mathbb{E}_{\mathbf{z}\sim q}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}\right]$$

So

$$\mathcal{F}(q,\theta) = \left\langle \log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z})} \right\rangle_q = \underset{\text{proposal}}{\mathbb{E}_{\mathbf{z}\sim q}}\left[\log p(\mathbf{x}) \underset{\text{importance weight}}{\frac{p(\mathbf{z}|\mathbf{x})}{q(\mathbf{z})}}\right]$$

-

Suggests more accurate importance sampling:

$$\ell(\theta) = \log \mathbb{E}_{\mathbf{z}_1\ldots\mathbf{z}_K \overset{\text{iid}}{\sim} q}\left[\frac{1}{K}\sum_k \frac{p(\mathbf{x},\mathbf{z}_k)}{q(\mathbf{z}_k)}\right] \geq \mathbb{E}_{\mathbf{z}_1\ldots\mathbf{z}_K \overset{\text{iid}}{\sim} q}\left[\log \frac{1}{K}\sum_k \frac{p(\mathbf{x},\mathbf{z}_k)}{q(\mathbf{z}_k)}\right]$$

Tighter bound, and reparametrisation friendly, but as $K \to \infty$ the signal for learning amortised q grows weaker so VAE learning doesn't always improve.

Normalising flows

$$\mathcal{F}(q, \theta) = \langle \log p(\mathbf{x}, \mathbf{z}|\theta) \rangle_q - \langle \log q(\mathbf{z}) \rangle_q$$

To evaluate $\mathcal{F}$ (or its gradients) we need to be able to find expectations wrt q (e.g. by Monte Carlo) and evaluate the log-density – usually restricts us to tractable inferential families.

Normalising flows

$$\mathcal{F}(q, \theta) = \langle \log p(\mathbf{x}, \mathbf{z}|\theta) \rangle_q - \langle \log q(\mathbf{z}) \rangle_q$$

To evaluate $\mathcal{F}$ (or its gradients) we need to be able to find expectations wrt q (e.g. by Monte Carlo) and evaluate the log-density – usually restricts us to tractable inferential families.

Consider defining a recognition model $q(\mathbf{z})$ implicitly by:

$$\mathbf{z}_0 \sim q_0(\cdot; \mathbf{x}) \qquad \leftarrow \text{ fixed, tractable, e.g. } \mathcal{N}(\mathbf{x}, I)$$

$$\mathbf{z} = f_K(f_{K-1}(\dots f_1(\mathbf{z}_0))) \qquad \leftarrow f_k \text{ smooth, invertible, parametrised by } \phi$$

Normalising flows

$$\mathcal{F}(q,\theta) = \langle \log p(\mathbf{x},\mathbf{z}|\theta)\rangle_q - \langle \log q(\mathbf{z})\rangle_q$$

To evaluate $\mathcal{F}$ (or its gradients) we need to be able to find expectations wrt q (e.g. by Monte Carlo) and evaluate the log-density – usually restricts us to tractable inferential families.

Consider defining a recognition model $q(\mathbf{z})$ implicitly by:

$$\mathbf{z}_0 \sim q_0(\cdot;\mathbf{x}) \qquad \leftarrow \text{ fixed, tractable, e.g. } \mathcal{N}(\mathbf{x}, I)$$

$$\mathbf{z} = f_K(f_{K-1}(\dots f_1(\mathbf{z}_0))) \qquad \leftarrow f_k \text{ smooth, invertible, parametrised by } \phi$$

Then we can both compute expectations under q and evaluate its log density:

$$\langle F(\mathbf{z})\rangle_q = \langle F(f_K(f_{K-1}(\dots f_1(\mathbf{z}_0))))\rangle_{q_0}$$

$$\log q(\mathbf{z}) = \log q_0(f_1^{-1}(f_2^{-1}(\dots f_K^{-1}(\mathbf{z})))) - \sum_k \log|\nabla f_k|$$

where the second result applies from repeated transformations of variables

$$\mathbf{z}_k = f_k(\mathbf{z}_{k-1}) \;\Rightarrow\; q(\mathbf{z}_k) = q(f_k^{-1}(\mathbf{z}_k))\left|\frac{\partial \mathbf{z}_{k-1}}{\partial \mathbf{z}_k}\right| = q(f_k^{-1}(\mathbf{z}_k))|\nabla f_k(\mathbf{z}_{k-1})|^{-1}$$

Normalising flows

So, given a sample $\mathbf{z}_0^s \overset{\text{iid}}{\sim} q_0(\cdot; \mathbf{x})$:

$$\mathcal{F}(\phi, \theta) \approx \frac{1}{S}\sum_s \log p(\mathbf{x}, f_K(\ldots f_1(\mathbf{z}_0^s)))) + \mathbf{H}[q_0] + \frac{1}{S}\sum_s \sum_k \log\left|\nabla f_k(f_{k-1}(\ldots f_1(\mathbf{z}_0^s)))\right|$$

and we can compute gradients of this expression wrt θ and ϕ.

Useful fs (from Rezende & Mohammed 2015):

$$f(\mathbf{z}) = \mathbf{z} + \mathbf{u}h(\mathbf{w}^\mathsf{T}\mathbf{z} + b) \qquad \Rightarrow |\nabla f| = \left|1 + \mathbf{u}^\mathsf{T}\psi(\mathbf{z})\right| \qquad \psi(\mathbf{z}) = h'(\mathbf{w}^\mathsf{T}\mathbf{z} + b)\mathbf{w}$$

$$f(\mathbf{z}) = \mathbf{z} + \frac{\beta}{\alpha + |\mathbf{z} - \mathbf{z}_0|} \qquad \Rightarrow |\nabla f| = [1 + \beta h]^{d-1}[1 + \beta h + \beta h' r]$$

$$r = |\mathbf{z} - \mathbf{z}_0|, h = \frac{1}{\alpha + r}$$

Both can be cascaded to give a flexible variational family.

Diffusion probabilistic models

Multi-stage flexible generative process (like normalising flow) with *fixed recognition* model.

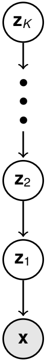
In our notation:

Diffusion probabilistic models

Multi-stage flexible generative process (like normalising flow) with *fixed recognition* model.

In our notation:

- Define observations $\mathbf{x}$ and latents $\mathbf{z}_1 \ldots \mathbf{z}_K$.



Diffusion probabilistic models

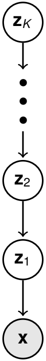
Multi-stage flexible generative process (like normalising flow) with *fixed recognition* model.

In our notation:

- Define observations $\mathbf{x}$ and latents $\mathbf{z}_1 \ldots \mathbf{z}_K$.
- Fix "diffusion" recognition model (the "forward" model)

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$

$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) = \mathcal{N}\left(\sqrt{1-\beta_k}\mathbf{z}_{k-1}, \beta_k \mathrm{I}\right)$$



Diffusion probabilistic models

Multi-stage flexible generative process (like normalising flow) with *fixed recognition* model.

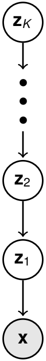
In our notation:

- Define observations $\mathbf{x}$ and latents $\mathbf{z}_1 \ldots \mathbf{z}_K$.
- Fix "diffusion" recognition model (the "forward" model)

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$
$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) = \mathcal{N}\left(\sqrt{1-\beta_k}\mathbf{z}_{k-1}, \beta_k \mathrm{I}\right)$$

- Parametrise generative model (the "backward" model)

$$p(\mathbf{z}_K) = \mathcal{N}(\mathbf{0}, \mathrm{I})$$
$$p(\mathbf{z}_{k-1}|\mathbf{z}_k;\theta) = \mathcal{N}(\mu_\theta(\mathbf{z}_k, k), \Sigma_\theta(\mathbf{z}_k, k))$$
$$p(\mathbf{x}|\mathbf{z}_1;\theta) = \mathcal{N}\big(\underbrace{\mu_\theta(\mathbf{z}_1, 1), \Sigma_\theta(\mathbf{z}_1, 1)}_{\text{usually NNs}}\big)$$



Diffusion probabilistic models

Multi-stage flexible generative process (like normalising flow) with *fixed recognition* model.

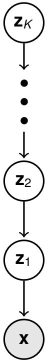
In our notation:

- Define observations $\mathbf{x}$ and latents $\mathbf{z}_1 \dots \mathbf{z}_K$.
- Fix "diffusion" recognition model (the "forward" model)

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$
$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) = \mathcal{N}\left(\sqrt{1-\beta_k}\mathbf{z}_{k-1}, \beta_k \mathrm{I}\right)$$

- Parametrise generative model (the "backward" model)

$$p(\mathbf{z}_K) = \mathcal{N}(\mathbf{0}, \mathrm{I})$$
$$p(\mathbf{z}_{k-1}|\mathbf{z}_k; \theta) = \mathcal{N}(\mu_\theta(\mathbf{z}_k, k), \Sigma_\theta(\mathbf{z}_k, k))$$
$$p(\mathbf{x}|\mathbf{z}_1; \theta) = \mathcal{N}\big(\underbrace{\mu_\theta(\mathbf{z}_1, 1), \Sigma_\theta(\mathbf{z}_1, 1)}_{\text{usually NNs}}\big)$$



Diffusion recognition sends $q(\mathbf{z}_K) \overset{K\to\infty}{\to} \mathcal{N}(\mathbf{0}, \mathrm{I})$.
In the limit $\beta_k \to 0$ the reciprocal normal generation is correct.

Diffusion probabilistic models

Multi-stage flexible generative process (like normalising flow) with *fixed recognition* model.

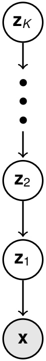
In our notation:

- Define observations $\mathbf{x}$ and latents $\mathbf{z}_1 \dots \mathbf{z}_K$.
- Fix "diffusion" recognition model (the "forward" model)

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$
$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) = \mathcal{N}\left(\sqrt{1-\beta_k}\mathbf{z}_{k-1}, \beta_k \mathrm{I}\right)$$

- Parametrise generative model (the "backward" model)

$$p(\mathbf{z}_K) = \mathcal{N}(\mathbf{0}, \mathrm{I})$$
$$p(\mathbf{z}_{k-1}|\mathbf{z}_k;\theta) = \mathcal{N}(\mu_\theta(\mathbf{z}_k, k), \Sigma_\theta(\mathbf{z}_k, k))$$
$$p(\mathbf{x}|\mathbf{z}_1;\theta) = \mathcal{N}\big(\underbrace{\mu_\theta(\mathbf{z}_1, 1), \Sigma_\theta(\mathbf{z}_1, 1)}_{\text{usually NNs}}\big)$$



Diffusion recognition sends $q(\mathbf{z}_K) \stackrel{K\to\infty}{\to} \mathcal{N}(\mathbf{0}, \mathrm{I})$.
In the limit $\beta_k \to 0$ the reciprocal normal generation is correct.
But as $\beta \to 0$ and $K \to \infty$ the link between observation and $\mathbf{z}_K$ becomes uninformative.

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$.

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$.
The diffusion assumption makes these marginals easy to compute.

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$. The diffusion assumption makes these marginals easy to compute.

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$. The diffusion assumption makes these marginals easy to compute.

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$

$$q(\mathbf{z}_2|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, \sqrt{1-\beta_2}(\beta_1 \mathrm{I})\sqrt{1-\beta_2} + \beta_2 \mathrm{I}\right)$$

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$. The diffusion assumption makes these marginals easy to compute.

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$

$$q(\mathbf{z}_2|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, \sqrt{1-\beta_2}(\beta_1 \mathrm{I})\sqrt{1-\beta_2} + \beta_2 \mathrm{I}\right)$$

$$= \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, (1 - (1-\beta_2)(1-\beta_1))\mathrm{I}\right)$$

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$. The diffusion assumption makes these marginals easy to compute.

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$

$$q(\mathbf{z}_2|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, \sqrt{1-\beta_2}(\beta_1 \mathrm{I})\sqrt{1-\beta_2} + \beta_2 \mathrm{I}\right)$$

$$= \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, (1 - (1-\beta_2)(1-\beta_1))\mathrm{I}\right)$$

Let $\alpha_k = 1 - \beta_k$ and $\bar{\alpha}_k = \prod_{i=1}^{k} \alpha_i$ and suppose

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1 - \bar{\alpha}_k)\mathrm{I}\right)$$

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$. The diffusion assumption makes these marginals easy to compute.

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$

$$q(\mathbf{z}_2|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, \sqrt{1-\beta_2}(\beta_1 \mathrm{I})\sqrt{1-\beta_2} + \beta_2 \mathrm{I}\right)$$

$$= \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, (1 - (1-\beta_2)(1-\beta_1))\mathrm{I}\right)$$

Let $\alpha_k = 1 - \beta_k$ and $\bar{\alpha}_k = \prod_{i=1}^{k} \alpha_i$ and suppose

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1 - \bar{\alpha}_k)\mathrm{I}\right)$$

$$\Rightarrow q(\mathbf{z}_{k+1}|\mathbf{x}) = \mathcal{N}\left(\sqrt{\alpha_{k+1}}\sqrt{\bar{\alpha}_k}\mathbf{x}, \alpha_{k+1}(1 - \bar{\alpha}_k)\mathrm{I} + \beta_{k_1}\mathrm{I}\right)$$

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1)\rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k)\rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K)\rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$. The diffusion assumption makes these marginals easy to compute.

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$

$$q(\mathbf{z}_2|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, \sqrt{1-\beta_2}(\beta_1 \mathrm{I})\sqrt{1-\beta_2} + \beta_2 \mathrm{I}\right)$$

$$= \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, (1 - (1-\beta_2)(1-\beta_1))\mathrm{I}\right)$$

Let $\alpha_k = 1 - \beta_k$ and $\bar{\alpha}_k = \prod_{i=1}^{k} \alpha_i$ and suppose

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1 - \bar{\alpha}_k)\mathrm{I}\right)$$

$$\Rightarrow q(\mathbf{z}_{k+1}|\mathbf{x}) = \mathcal{N}\left(\sqrt{\alpha_{k+1}}\sqrt{\bar{\alpha}_k}\mathbf{x}, \alpha_{k+1}(1 - \bar{\alpha}_k)\mathrm{I} + \beta_{k_1}\mathrm{I}\right)$$

$$= \mathcal{N}\left(\sqrt{\bar{\alpha}_{k+1}}\mathbf{x}, \alpha_{k+1}(1 - \bar{\alpha}_k)\mathrm{I} + \beta_{k_1}\mathrm{I}\right)$$

Diffusion models

Free energy

$$\mathcal{F} = \left\langle \log p(\mathbf{x}|\mathbf{z}_1) + \sum_{k=2}^{K} \log p(\mathbf{z}_{k-1}|\mathbf{z}_K) + \log p(\mathbf{z}_k) \right\rangle_{q(\mathbf{z}_{1:K}|\mathbf{x})} - \mathbf{H}[q(\mathbf{z}_{1:K}|\mathbf{x})]$$

$$= \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

So learning requires expectations (usually based on samples) under $q(\mathbf{z}_k)$ and $q(\mathbf{z}_{k-1}|\mathbf{z}_k)$. The diffusion assumption makes these marginals easy to compute.

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_1}\mathbf{x}, \beta_1 \mathrm{I}\right)$$

$$q(\mathbf{z}_2|\mathbf{x}) = \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, \sqrt{1-\beta_2}(\beta_1 \mathrm{I})\sqrt{1-\beta_2} + \beta_2 \mathrm{I}\right)$$

$$= \mathcal{N}\left(\sqrt{1-\beta_2}\sqrt{1-\beta_1}\mathbf{x}, (1 - (1-\beta_2)(1-\beta_1))\mathrm{I}\right)$$

Let $\alpha_k = 1 - \beta_k$ and $\bar{\alpha}_k = \prod_{i=1}^{k} \alpha_i$ and suppose

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1 - \bar{\alpha}_k)\mathrm{I}\right)$$

$$\Rightarrow q(\mathbf{z}_{k+1}|\mathbf{x}) = \mathcal{N}\left(\sqrt{\alpha_{k+1}}\sqrt{\bar{\alpha}_k}\mathbf{x}, \alpha_{k+1}(1 - \bar{\alpha}_k)\mathrm{I} + \beta_{k_1}\mathrm{I}\right)$$

$$= \mathcal{N}\left(\sqrt{\bar{\alpha}_{k+1}}\mathbf{x}, \alpha_{k+1}(1 - \bar{\alpha}_k)\mathrm{I} + \beta_{k_1}\mathrm{I}\right)$$

demonstrating the premise by recursion.

Diffusion models

$$\mathcal{F} = \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k, \mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1 - \bar{\alpha}_k)\mathrm{I}\right)$$

Diffusion models

$$\mathcal{F} = \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k, \mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1 - \bar{\alpha}_k)\mathrm{I}\right)$$

Now,

$$q(\mathbf{z}_{k-1}|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{k-1}}\mathbf{x}, (1 - \bar{\alpha}_{k-1})\mathrm{I}\right)$$

$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) \propto \mathcal{N}\left(\mathbf{z}_{k-1}; \frac{1}{\sqrt{1-\beta_k}}\mathbf{z}_K, \frac{\beta_k}{1-\beta_k} I\right)$$

Diffusion models

$$\mathcal{F} = \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1 - \bar{\alpha}_k)\mathrm{I}\right)$$

Now,

$$q(\mathbf{z}_{k-1}|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{k-1}}\mathbf{x}, (1 - \bar{\alpha}_{k-1})\mathrm{I}\right)$$

$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) \propto \mathcal{N}\left(\mathbf{z}_{k-1}; \frac{1}{\sqrt{1-\beta_k}}\mathbf{z}_K, \frac{\beta_k}{1-\beta_k} I\right)$$

$$\Rightarrow q(\mathbf{z}_{k-1}|\mathbf{z}_k, \mathbf{x}) = \mathcal{N}\left(\frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k}\mathbf{x} + \frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\mathbf{z}_k, \frac{\beta_k(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\right)$$

Diffusion models

$$\mathcal{F} = \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1-\bar{\alpha}_k)\mathrm{I}\right)$$

Now,

$$q(\mathbf{z}_{k-1}|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{k-1}}\mathbf{x}, (1-\bar{\alpha}_{k-1})\mathrm{I}\right)$$

$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) \propto \mathcal{N}\left(\mathbf{z}_{k-1}; \frac{1}{\sqrt{1-\beta_k}}\mathbf{z}_K, \frac{\beta_k}{1-\beta_k} I\right)$$

$$\Rightarrow q(\mathbf{z}_{k-1}|\mathbf{z}_k, \mathbf{x}) = \mathcal{N}\left(\frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k}\mathbf{x} + \frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\mathbf{z}_k, \frac{\beta_k(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\right)$$

- So $\langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_{k-1},\mathbf{z}_k|\mathbf{x})}$ can be computed by sampling from $\mathbf{z}_k$ and using (closed form) conditional for $q(\mathbf{z}_{k-1}|\mathbf{z}_k, \mathbf{x})$.

Diffusion models

$$\mathcal{F} = \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1-\bar{\alpha}_k)\mathrm{I}\right)$$

Now,

$$q(\mathbf{z}_{k-1}|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{k-1}}\mathbf{x}, (1-\bar{\alpha}_{k-1})\mathrm{I}\right)$$

$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) \propto \mathcal{N}\left(\mathbf{z}_{k-1}; \frac{1}{\sqrt{1-\beta_k}}\mathbf{z}_K, \frac{\beta_k}{1-\beta_k} I\right)$$

$$\Rightarrow q(\mathbf{z}_{k-1}|\mathbf{z}_k, \mathbf{x}) = \mathcal{N}\left(\frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k}\mathbf{x} + \frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\mathbf{z}_k, \frac{\beta_k(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\right)$$

- So $\langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_{k-1},\mathbf{z}_k|\mathbf{x})}$ can be computed by sampling from $\mathbf{z}_k$ and using (closed form) conditional for $q(\mathbf{z}_{k-1}|\mathbf{z}_k, \mathbf{x})$.
- Reparametrisation (as in the VAE) makes it possible to also optimise β_k.

Diffusion models

$$\mathcal{F} = \langle \log p(\mathbf{x}|\mathbf{z}_1) \rangle_{q(\mathbf{z}_1|\mathbf{x})} + \sum_{k=2}^{K} \langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_k,\mathbf{z}_{k-1}|\mathbf{x})} + \langle \log p(\mathbf{z}_K) \rangle_{q(\mathbf{z}_K|\mathbf{x})} - \sum_{k=1}^{K} \mathbf{H}[\cdot]$$

$$q(\mathbf{z}_k|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\mathbf{x}, (1 - \bar{\alpha}_k)\mathrm{I}\right)$$

Now,

$$q(\mathbf{z}_{k-1}|\mathbf{x}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{k-1}}\mathbf{x}, (1 - \bar{\alpha}_{k-1})\mathrm{I}\right)$$

$$q(\mathbf{z}_k|\mathbf{z}_{k-1}) \propto \mathcal{N}\left(\mathbf{z}_{k-1}; \frac{1}{\sqrt{1-\beta_k}}\mathbf{z}_K, \frac{\beta_k}{1-\beta_k} I\right)$$

$$\Rightarrow q(\mathbf{z}_{k-1}|\mathbf{z}_k, \mathbf{x}) = \mathcal{N}\left(\frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k}\mathbf{x} + \frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\mathbf{z}_k, \frac{\beta_k(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}\right)$$

- So $\langle \log p(\mathbf{z}_{k-1}|\mathbf{z}_k) \rangle_{q(\mathbf{z}_{k-1},\mathbf{z}_k|\mathbf{x})}$ can be computed by sampling from $\mathbf{z}_k$ and using (closed form) conditional for $q(\mathbf{z}_{k-1}|\mathbf{z}_k, \mathbf{x})$.
- Reparametrisation (as in the VAE) makes it possible to also optimise β_k.
- Considerable recent work: noise-target NNs; conditional models; score-based diffusions ....

DDC Helmholtz machine

A (loosely) neurally inspired idea. Define q as an unnormalisable exponential family with a large set of sufficient statistics

$$q(\mathbf{z}) \propto e^{\sum_i \eta_i \psi_i(\mathbf{z})}$$

and parametrise by mean parameters $\boldsymbol{\mu} = \langle \boldsymbol{\psi}(\mathbf{z}) \rangle$: Distributed distributional code (DDC).

Train recognition model using sleep samples:

$$\boldsymbol{\mu} = \langle \boldsymbol{\psi}(\mathbf{z}) \rangle_q = f(\mathbf{x}; \phi)$$

$$\Delta\phi \propto \sum_s (\boldsymbol{\psi}(\mathbf{z}_s) - f(\mathbf{x}_s; \phi)) \nabla_\phi f(\mathbf{x}_s; \phi)$$

Also learn linear approximation $\nabla \log p(\mathbf{x}, \mathbf{z}|\theta) \approx A\boldsymbol{\psi}(\mathbf{z})$

$$A = \left(\sum_s \nabla \log p(\mathbf{x}_s, \mathbf{z}_s|\theta) \boldsymbol{\psi}(\mathbf{z}_s) \right)^{\mathsf{T}} \left(\sum_s \boldsymbol{\psi}(\mathbf{z}_s) \boldsymbol{\psi}(\mathbf{z}_s)^{\mathsf{T}} \right)^{-1}$$

Then

$$\langle \nabla \log p(\mathbf{x}, \mathbf{z}) \rangle_q \approx A \langle \boldsymbol{\psi}(\mathbf{z}) \rangle_q \approx Af(\mathbf{x}, \phi)$$

Approach can be generalised to an infinite dimensional $\boldsymbol{\psi}$ using the kernel trick.

Amortised Learning

If we aren't actually interested in inference, we can short-circuit general recognition and compute expectations for learning directly.

$$\nabla_\theta \ell(\theta) = \partial_\theta \mathcal{F}(q^*, \theta) = \partial_\theta \langle \log p(\mathcal{X}, \mathcal{Z}|\theta) \rangle_{q^*} = \langle \partial_\theta \log p(\mathcal{X}, \mathcal{Z}|\theta) \rangle_{p(\mathcal{Z}|\mathcal{X}, \theta)}$$

Amortised Learning

If we aren't actually interested in inference, we can short-circuit general recognition and compute expectations for learning directly.

$$\nabla_\theta \ell(\theta) = \partial_\theta \mathcal{F}(q^*, \theta) = \partial_\theta \langle \log p(\mathcal{X}, \mathcal{Z}|\theta) \rangle_{q^*} = \langle \partial_\theta \log p(\mathcal{X}, \mathcal{Z}|\theta) \rangle_{p(\mathcal{Z}|\mathcal{X},\theta)}$$

Suggests a wake-sleep approach:

- Sample $\{\mathbf{x}_s, \mathbf{z}_s\} \sim p(\mathcal{X}, \mathcal{Z}|\theta^k)$.
- Train regressor $\hat{J}_{\theta^k} : \mathbf{x}_s \mapsto \nabla_\theta \log p(\mathbf{x}_s, \mathbf{z}_s|\theta)|_{\theta^k}$
(or, for specific regressors, $\mapsto \log p(\mathbf{x}_s, \mathbf{z}_s|\theta^k)$ and differentiate prediction)
- Set $\theta^{k+1} = \theta^k + \alpha \sum_i \hat{J}_{\theta^k}(\mathbf{x}_i)$
(or $= \theta^k + \alpha \sum_i \nabla_\theta \hat{J}_\theta(\mathbf{x}_i)|_{\theta^k}$).

Derivative form works for (kernel/GP) regression for which regressor is linear in targets.

Amortised Learning

If we aren't actually interested in inference, we can short-circuit general recognition and compute expectations for learning directly.

$$\nabla_\theta \ell(\theta) = \partial_\theta \mathcal{F}(q^*, \theta) = \partial_\theta \langle \log p(\mathcal{X}, \mathcal{Z}|\theta) \rangle_{q^*} = \langle \partial_\theta \log p(\mathcal{X}, \mathcal{Z}|\theta) \rangle_{p(\mathcal{Z}|\mathcal{X},\theta)}$$

Suggests a wake-sleep approach:

- Sample $\{\mathbf{x}_s, \mathbf{z}_s\} \sim p(\mathcal{X}, \mathcal{Z}|\theta^k)$.
- Train regressor $\hat{J}_{\theta^k} : \mathbf{x}_s \mapsto \nabla_\theta \log p(\mathbf{x}_s, \mathbf{z}_s|\theta)|_{\theta^k}$
  (or, for specific regressors, $\mapsto \log p(\mathbf{x}_s, \mathbf{z}_s|\theta^k)$ and differentiate prediction)
- Set $\theta^{k+1} = \theta^k + \alpha \sum_i \hat{J}_{\theta^k}(\mathbf{x}_i)$
  (or $= \theta^k + \alpha \sum_i \nabla_\theta \hat{J}_\theta(\mathbf{x}_i)|_{\theta^k}$).

Derivative form works for (kernel/GP) regression for which regressor is linear in targets.

For conditional exponential family models

$$\log p(\mathcal{X}, \mathcal{Z}|\theta) = \eta(\mathbf{z}, \theta)^{\mathsf{T}} \mathbf{T}(\mathbf{x}) - \Phi(\mathbf{z}, \theta) + \log p(\mathbf{z}|\theta)$$

$$\Rightarrow \langle \log p(\mathcal{X}, \mathcal{Z}|\theta) \rangle_{q^*} = \langle \eta(\mathbf{z}, \theta) \rangle_{q^*}^{\mathsf{T}} \mathbf{T}(\mathbf{x}) - \langle \Phi(\mathbf{z}, \theta) + \log p(\mathbf{z}|\theta) \rangle_{q^*}$$

and regressors can be trained to functions of $\mathbf{z}$ alone, with $T(\mathbf{x})$ then evaluated on (wake-phase) data.

Generative models

In practice, much of the VAE and related work has used a common generative model:

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, I)$$
$$\mathbf{x} \sim \mathcal{N}(\mathbf{g}(\mathbf{z}; \boldsymbol{\theta}), \psi I)$$

where g is a neural network.

- Overcomplicated: if $\dim(\mathbf{z})$ is large enough the optimal solution has $\psi \to 0$, $q(\mathbf{z}; \mathbf{x}) \to \delta(\mathbf{z} - f(\mathbf{x}, \phi))$. In effect, the generative model learns a flow to transform a normal density to the target.
- Oversimplified: if $\dim(\mathbf{z})$ is small, this is just non-linear PCA!

Interesting latent representations are likely to require more structured generative models. Recent work has approached such models in both VAE and DDC frameworks.

Structured VAEs

Consider a model where $p(\mathcal{Z}|\theta)$ has tractable joint exponential-family potentials and

$$p(\mathcal{X}|\mathcal{Z},\Gamma) = \prod_i p(\mathbf{x}_i|\mathbf{z}_i,\gamma_i)$$

are intractable (say neural net + normal) cond ind observations. γ_i might be the same for all i.

Structured VAEs

Consider a model where $p(\mathcal{Z}|\theta)$ has tractable joint exponential-family potentials and

$$p(\mathcal{X}|\mathcal{Z}, \Gamma) = \prod_i p(\mathbf{x}_i|\mathbf{z}_i, \gamma_i)$$

are intractable (say neural net + normal) cond ind observations. γ_i might be the same for all i.

Consider factored variational inference $q(\mathcal{Z}) = \prod_i q_i(\mathbf{z}_i)$. With no further constraint,

$$\begin{aligned}\log q_i^*(\mathbf{z}_i) &\underset{+C}{=} \langle \log p(\mathcal{Z}, \mathcal{X}) \rangle_{q_{\neg i}} \underset{+C}{=} \langle \log p(\mathbf{z}_i|\mathcal{Z}_{\neg i}) + \log p(\mathbf{x}_i|\mathbf{z}_i) \rangle_{q_{\neg i}} \\ &\underset{+C}{=} \langle \boldsymbol{\eta}_{\neg i} \rangle_{q_{\neg i}}^{\mathsf{T}} \boldsymbol{\psi}_i(\mathbf{z}_i) + \log p(\mathbf{x}_i|\mathbf{z}_i)\end{aligned}$$

where we have exploited the exponential-family form of $p(\mathcal{Z})$. $\boldsymbol{\psi}_i$ are effective suff stats – including log normalisers of children in a DAG; $\boldsymbol{\eta}_{\neg i}$ is a function of $\mathcal{Z}_{\neg i}$.

Structured VAEs

Consider a model where $p(\mathcal{Z}|\theta)$ has tractable joint exponential-family potentials and

$$p(\mathcal{X}|\mathcal{Z},\Gamma) = \prod_i p(\mathbf{x}_i|\mathbf{z}_i,\gamma_i)$$

are intractable (say neural net + normal) cond ind observations. γ_i might be the same for all i.

Consider factored variational inference $q(\mathcal{Z}) = \prod_i q_i(\mathbf{z}_i)$. With no further constraint,

$$\begin{aligned}\log q_i^*(\mathbf{z}_i) &\underset{+C}{=} \langle \log p(\mathcal{Z},\mathcal{X})\rangle_{q_{\neg i}} \underset{+C}{=} \langle \log p(\mathbf{z}_i|\mathcal{Z}_{\neg i}) + \log p(\mathbf{x}_i|\mathbf{z}_i)\rangle_{q_{\neg i}} \\ &\underset{+C}{=} \langle \boldsymbol{\eta}_{\neg i}\rangle_{q_{\neg i}}^{\mathsf{T}} \boldsymbol{\psi}_i(\mathbf{z}_i) + \log p(\mathbf{x}_i|\mathbf{z}_i)\end{aligned}$$

where we have exploited the exponential-family form of $p(\mathcal{Z})$. $\boldsymbol{\psi}_i$ are effective suff stats – including log normalisers of children in a DAG; $\boldsymbol{\eta}_{\neg i}$ is a function of $\mathcal{Z}_{\neg i}$.

Now, choose the parametric form $q_i(\mathbf{z}_i) = e^{\tilde{\boldsymbol{\eta}}_i^{\mathsf{T}} \boldsymbol{\psi}_i(\mathbf{z}_i) - \Phi_i(\tilde{\boldsymbol{\eta}}_i)}$. Constrained optimum has form

$$\log q_i^*(\mathbf{z}_i) \underset{+C}{=} \langle \boldsymbol{\eta}_{\neg i}\rangle_{q_{\neg i}}^{\mathsf{T}} \boldsymbol{\psi}_i(\mathbf{z}_i) + \boldsymbol{\rho}(\mathbf{x}_i)^{\mathsf{T}} \boldsymbol{\psi}_i(\mathbf{z}_i)$$

for some $\mathbf{x}_i$-dependent natural parameter. Introduce recognition models:

$$\boldsymbol{\rho}(\mathbf{x}_i) = f_i(\mathbf{x}_i, \phi_i)$$

Recognition function f_i might be same for all i if all likelihoods are the same (*e.g.* HMM).

Structured VAE learning

Now, the free-energy can be written as a function of parameters and recognition parameters:

$$\mathcal{F}(\theta, \Gamma, \{\phi_i\}) = \left\langle \sum_i \log p(\mathbf{x}_i|\mathbf{z}_i, \gamma_i) + \log p(\mathcal{Z}|\theta) \right\rangle_{q(\mathcal{Z};\theta,\{\phi_i\})} + \sum_i \mathbf{H}[q_i]$$

Structured VAE learning

Now, the free-energy can be written as a function of parameters and recognition parameters:

$$\mathcal{F}(\theta, \Gamma, \{\phi_i\}) = \left\langle \sum_i \log p(\mathbf{x}_i|\mathbf{z}_i, \gamma_i) + \log p(\mathcal{Z}|\theta) \right\rangle_{q(\mathcal{Z};\theta,\{\phi_i\})} + \sum_i \mathbf{H}[q_i]$$

$$= \sum_i \underbrace{\langle \log p(\mathbf{x}_i|\mathbf{z}_i, \gamma_i)\rangle_{q_i(\mathbf{z}_i;\theta,\phi_i)} + \mathbf{H}[q_i]}_{\mathcal{F}_i} + \langle \log p(\mathcal{Z}|\theta)\rangle_{q(\mathcal{Z};\theta,\{\phi_i\})}$$

Structured VAE learning

Now, the free-energy can be written as a function of parameters and recognition parameters:

$$\mathcal{F}(\theta, \Gamma, \{\phi_i\}) = \left\langle \sum_i \log p(\mathbf{x}_i|\mathbf{z}_i, \gamma_i) + \log p(\mathcal{Z}|\theta) \right\rangle_{q(\mathcal{Z};\theta,\{\phi_i\})} + \sum_i \mathbf{H}[q_i]$$

$$= \sum_i \underbrace{\langle \log p(\mathbf{x}_i|\mathbf{z}_i, \gamma_i)\rangle_{q_i(\mathbf{z}_i;\theta,\phi_i)} + \mathbf{H}[q_i]}_{\mathcal{F}_i} + \langle \log p(\mathcal{Z}|\theta)\rangle_{q(\mathcal{Z};\theta,\{\phi_i\})}$$

Updates on θ are just as for tractable model.

Structured VAE learning

Now, the free-energy can be written as a function of parameters and recognition parameters:

$$\mathcal{F}(\theta, \Gamma, \{\phi_i\}) = \left\langle \sum_i \log p(\mathbf{x}_i|\mathbf{z}_i, \gamma_i) + \log p(\mathcal{Z}|\theta) \right\rangle_{q(\mathcal{Z};\theta,\{\phi_i\})} + \sum_i \mathbf{H}[q_i]$$

$$= \sum_i \underbrace{\langle \log p(\mathbf{x}_i|\mathbf{z}_i, \gamma_i) \rangle_{q_i(\mathbf{z}_i;\theta,\phi_i)} + \mathbf{H}[q_i]}_{\mathcal{F}_i} + \langle \log p(\mathcal{Z}|\theta) \rangle_{q(\mathcal{Z};\theta,\{\phi_i\})}$$

Updates on θ are just as for tractable model.

To update each ϕ_i and γ_i, find $\langle \boldsymbol{\eta}_{\neg i} \rangle_{q_{\neg i}}$ to give the "prior". Generate reparametrised samples $\mathbf{z}_i^s \sim q_i$. Then

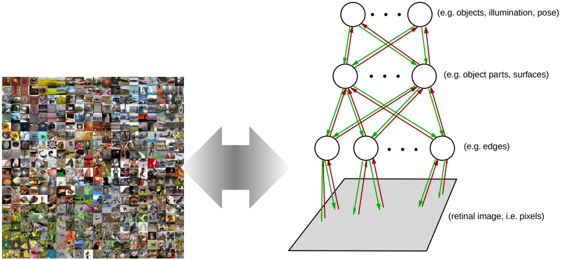
$$\frac{\partial}{\partial \gamma_i} \mathcal{F}_i = \sum_s \nabla_{\gamma_i} \log p(\mathbf{x}_i, \mathbf{z}_i^s; \gamma_i)$$

$$\frac{\partial}{\partial \phi_i} \mathcal{F}_i = \sum_s \frac{\partial}{\partial \mathbf{z}_i^s} (\log p(\mathbf{x}_i, \mathbf{z}_i^s; \gamma_i) - \log q(\mathbf{z}_i^s; \mathbf{f}(\mathbf{x}_i))) \frac{d\mathbf{z}_i^s}{d\phi} + \frac{\partial}{\partial \mathbf{f}(\mathbf{x}_i)} \log q(\mathbf{z}_i^s; \mathbf{f}(\mathbf{x}_i)) \frac{d\mathbf{f}(\mathbf{x}_i)}{d\phi}$$

as for the standard VAE.

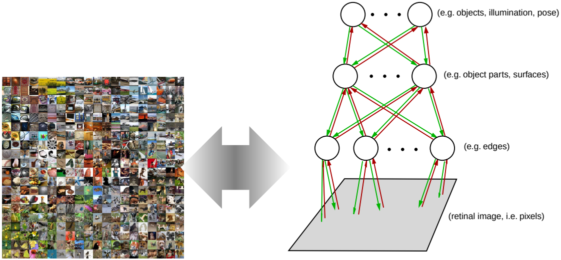
Likelihoods

An explicit generative likelihood (or energy) seems essential to match model to data



Likelihoods

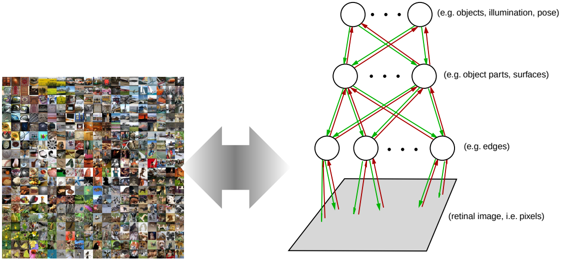
An explicit generative likelihood (or energy) seems essential to match model to data



. . . but introduces challenges

Likelihoods

An explicit generative likelihood (or energy) seems essential to match model to data

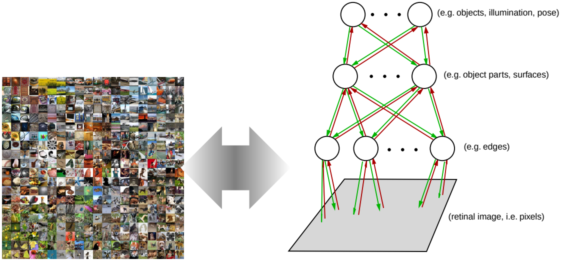


. . . but introduces challenges

- tractability: difficulty inverting non-linear generation creates bias

Likelihoods

An explicit generative likelihood (or energy) seems essential to match model to data

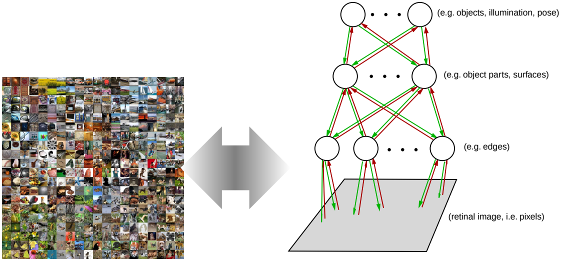


. . . but introduces challenges

- tractability: difficulty inverting non-linear generation creates bias
- relevance: irrelevant features must be modelled

Likelihoods

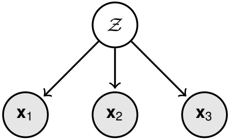
An explicit generative likelihood (or energy) seems essential to match model to data



. . . but introduces challenges

- tractability: difficulty inverting non-linear generation creates bias
- relevance: irrelevant features must be modelled
- distributional choices: noise models may be inaccurate

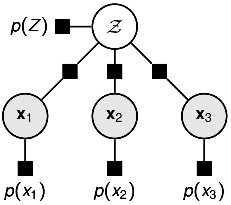
Recognition parametrisation



$$p(\mathcal{X}, \mathcal{Z}) = p(\mathcal{Z}) \prod_j p(\mathbf{x}_j | \mathcal{Z})$$

- **Start with a conventional generative model:**

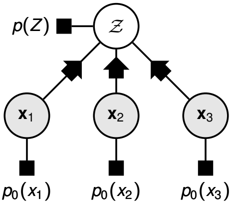
Recognition parametrisation



$$p(\mathcal{X}, \mathcal{Z}) = p(\mathcal{Z}) \prod_j p(\mathbf{x}_j) \frac{p(\mathcal{Z}|\mathbf{x}_j)}{p(\mathcal{Z})}$$

- **Start with a conventional generative model:**
  - recognition can be defined by Bayes rule

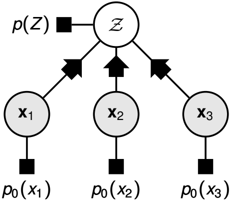
Recognition parametrisation



$$\mathsf{P}_{\theta,\mathbb{X}^{(N)}}(\mathcal{X},\mathcal{Z}) = p(\mathcal{Z}) \prod_j p_0(\mathbf{x}_j) \frac{f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)}{F_{\theta j}(\mathcal{Z})}$$

- **Recognition-parametrised model (RPM):**
  - $p_0(\mathbf{x}_j)$ set to a *non-parametric* marginal, e.g. $\frac{1}{N}\sum \delta(\mathbf{x}_j - \mathbf{x}_j^{(n)})$ (no learnt parameters)

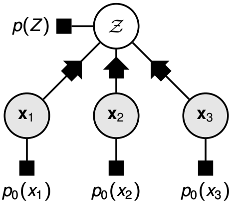
Recognition parametrisation



$$\mathrm{P}_{\theta,\mathbb{X}^{(N)}}(\mathcal{X},\mathcal{Z}) = p(\mathcal{Z}) \prod_j p_0(\mathbf{x}_j) \frac{f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)}{F_{\theta j}(\mathcal{Z})}$$

- **Recognition-parametrised model (RPM):**
  - $p_0(\mathbf{x}_j)$ set to a *non-parametric* marginal, e.g. $\frac{1}{N}\sum \delta(\mathbf{x}_j - \mathbf{x}_j^{(n)})$ (no learnt parameters)
  - $f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)$ a parametrised recognition factor, non-linear and conjugate to $p(\mathcal{Z})$

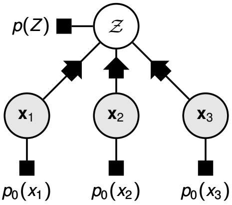
Recognition parametrisation



$$\mathrm{P}_{\theta,\mathbb{X}^{(N)}}(\mathcal{X},\mathcal{Z}) = p(\mathcal{Z})\prod_j p_0(\mathbf{x}_j)\frac{f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)}{F_{\theta j}(\mathcal{Z})}$$

- **Recognition-parametrised model (RPM):**
  - $p_0(\mathbf{x}_j)$ set to a *non-parametric* marginal, e.g. $\frac{1}{N}\sum\delta(\mathbf{x}_j - \mathbf{x}_j^{(n)})$ (no learnt parameters)
  - $f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)$ a parametrised recognition factor, non-linear and conjugate to $p(\mathcal{Z})$
  - $F_{\theta j}(\mathcal{Z}) = \int d\mathbf{x}_j\, p_o(\mathbf{x}_j) f_{\theta j}(\mathcal{Z}|\mathbf{x}_j) = \dfrac{1}{N}\sum f_{\theta j}(\mathcal{Z}|\mathbf{x}_j^{(n)})$
    (fully determined by $f_{\theta j}$ parameters and $p_0(\mathbf{x}_j)$)

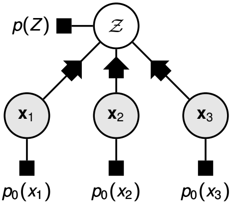
Recognition parametrisation



$$\mathrm{P}_{\theta,\mathbb{X}^{(N)}}(\mathcal{X},\mathcal{Z}) = p(\mathcal{Z}) \prod_j p_0(\mathbf{x}_j) \frac{f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)}{F_{\theta j}(\mathcal{Z})}$$

- **Recognition-parametrised model (RPM):**
  - $p_0(\mathbf{x}_j)$ set to a *non-parametric* marginal, e.g. $\frac{1}{N}\sum \delta(\mathbf{x}_j - \mathbf{x}_j^{(n)})$ (no learnt parameters)
  - $f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)$ a parametrised recognition factor, non-linear and conjugate to $p(\mathcal{Z})$
  - $F_{\theta j}(\mathcal{Z}) = \int d\mathbf{x}_j \, p_o(\mathbf{x}_j) f_{\theta j}(\mathcal{Z}|\mathbf{x}_j) = \frac{1}{N}\sum f_{\theta j}(\mathcal{Z}|\mathbf{x}_j^{(n)})$ (fully determined by $f_{\theta j}$ parameters and $p_0(\mathbf{x}_j)$)
- Properly normalised, but data-dependent (semi-parametric) model
- Likelihood optimised by variational (EM) methods

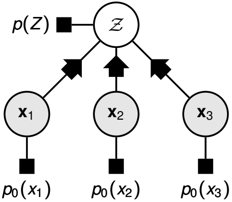
Recognition parametrised models



$$\mathsf{P}_{\theta,\mathbb{X}^{(N)}}(\mathcal{X},\mathcal{Z}) = p(\mathcal{Z}) \prod_j p_0(\mathbf{x}_j) \frac{f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)}{F_{\theta j}(\mathcal{Z})}$$

- no parametrised model of individual observed variables
- joint model focuses on capturing statistical dependence
- no explicit generation; likelihood found from recognition model alone
- consistent even with arbitrary nonlinearities!

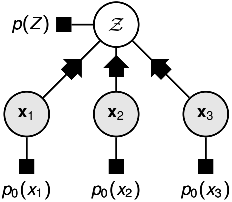
Recognition parametrised models



$$\mathsf{P}_{\theta,\mathbb{X}^{(N)}}(\mathcal{X},\mathcal{Z}) = p(\mathcal{Z}) \prod_j p_0(\mathbf{x}_j) \frac{f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)}{F_{\theta j}(\mathcal{Z})}$$

- no parametrised model of individual observed variables
- joint model focuses on capturing statistical dependence
- no explicit generation; likelihood found from recognition model alone
- consistent even with arbitrary nonlinearities!

Recognition parametrised models



$$\mathsf{P}_{\theta,\mathbb{X}^{(N)}}(\mathcal{X},\mathcal{Z}) = p(\mathcal{Z}) \prod_j p_0(\mathbf{x}_j) \frac{f_{\theta j}(\mathcal{Z}|\mathbf{x}_j)}{F_{\theta j}(\mathcal{Z})}$$

- no parametrised model of individual observed variables
- joint model focuses on capturing statistical dependence
- no explicit generation; likelihood found from recognition model alone
- consistent even with arbitrary nonlinearities!

. . . just a brief survey of a subset of current ideas.

A few things we hope you've learned in this course . . .

A few things we hope you've learned in this course . . .

- Exponential families are your friends.

A few things we hope you've learned in this course ...

- Exponential families are your friends.
- Latent variable models and conditional independence to uncover structured representations.

A few things we hope you've learned in this course . . .

- Exponential families are your friends.
- Latent variable models and conditional independence to uncover structured representations.
- Free-energies, maximum likelihood, variational approximation theory and variational Bayes.

A few things we hope you've learned in this course . . .

- Exponential families are your friends.
- Latent variable models and conditional independence to uncover structured representations.
- Free-energies, maximum likelihood, variational approximation theory and variational Bayes.
- Message passing exploits conditional independence.

A few things we hope you've learned in this course . . .

- Exponential families are your friends.
- Latent variable models and conditional independence to uncover structured representations.
- Free-energies, maximum likelihood, variational approximation theory and variational Bayes.
- Message passing exploits conditional independence.
- A rich toolkit of approximations, that you can compose in novel and useful ways.

A few things we hope you've learned in this course . . .

- Exponential families are your friends.
- Latent variable models and conditional independence to uncover structured representations.
- Free-energies, maximum likelihood, variational approximation theory and variational Bayes.
- Message passing exploits conditional independence.
- A rich toolkit of approximations, that you can compose in novel and useful ways.
- A theory of many approximations that helps ensure you understand their use and limitations (and may help derive new approaches).