Variational Noise-Contrastive Estimation

Benjamin Rhodes

Master of Science by Research
— Centre for Doctoral Training in Data Science —
Department of Informatics
University of Edinburgh

August 2018

Abstract

Unnormalised latent variable models represent a broad, highly flexible family of distributions. However, learning the parameters of such models is 'doubly-intractable' and therefore not amenable to MLE. This thesis proposes a new method for training unnormalised latent variable models that combines noise-contrastive estimation (NCE) with variational inference.

The method, which we call variational noise-contrastive estimation (VNCE), optimises a variational lower bound to the NCE objective function. We provide theoretical guarantees that VNCE is equivalent to NCE under certain assumptions and provide theory linking VNCE to standard variational inference. These contributions are validated on simple latent variable models.

Using VNCE, we provide a probabilistically-principled approach to an outstanding problem in the graphical models literature. Specifically, we learn a truncated Gaussian graphical model from incomplete data, treating the missing values as latent variables. In doing so, we develop an approximate algorithm that can leverage inference networks to learn with missing data, offering a novel solution to a recent problem in the variational inference literature.

List of figures

Chapter 3

Contribution

3.1 A gap in the literature

Both latent variable models and unnormalised models have received significant amounts of attention in the machine learning literature [46]. In theory, combining the two types of models should lead to more powerful density estimators, which can capture the structure in very complex data-generating distributions. However, as discussed in Section 2.1, amongst the multiple estimation techniques available for unnormalised models, only (persistent) CD is applicable to latent variable models RBM

The existence of only one estimation method for unnormalised latent variable models is undesirable for multiple reasons. Firstly, contrastive divergence is primarily designed for models for which we know the true posterior over latent variables. In theory, it is possible to combine contrastive divergence with variational inference—as discussed in Section 2.5.3—but there are few successful applications of this strategy in the literature. Secondly, contrastive divergence is not always the best choice of unnormalised estimation procedure, depending on the model, dataset and task, other choices such as score matching or NCE are preferable.

Developing a latent variable version for NCE is particularly appealing since NCE has the same level of generality as maximum likelihood estimation, whereas techniques like ratio matching and score matching are restricted to certain types of data. Moreover, the quality of estimation with NCE improves with the development of better methods for constructing a noise distribution. Recent work has shown this is possible [11], and we expect further advances can be made.

There has been essentially no work on generalising NCE to handle latent variables. Matsuda and Hyvarinen [34] derive a variant of NCE that works for finite mixture

models. Whilst such mixtures can be viewed as latent variable models, summing over these latents is an inexpensive operation. Hence, Matsuda and Hyvarinen[34] apply NCE to this sum without explicitly making use of the latent variable representation. Their approach is not designed, therefore, to handle more general latent variable models with intractable posteriors. Goodfellow et al[15] briefly mention that the standard variational lower bound to the log-likelihood can be used to get a lower bound on the *first*, but not second, term of the NCE objective function. Unable to handle this second term, they abandon the idea of combining NCE with variational inference.

3.2 Our contribution

This thesis presents four main contributions:

- VNCE: An extension of noise-contrastive estimation that can handle latent variables.
- A theoretical analysis of VNCE, illustrating its relationship to NCE, combined with empirical validation of the theory on toy problems.
- A novel method, called Cumulative Data Imputation (CDI), for applying variational inference to missing-data problems.
- Simulations validating the combined effectiveness of VNCE and CDI on an outstanding problem from the graphical models literature [30].

The first, and primary, contribution of this thesis is to extend NCE to handle latent variable models. Our approach combines NCE with variational inference by deriving a variational lower bound on the NCE objective, overcoming a key obstacle recognised by Goodfellow et al. [15].

Given this lower bound, we derive theoretical results paralleling those for standard variational inference. We show that optimising this lower bound with respect to a variational distribution is a valid form of approximate inference, although no longer based on minimising a KL-divergence, but a different kind of divergence. Moreover, we derive an EM-type algorithm for VNCE that provably never decreases the objective function. These theoretical contributions are then empirically validated. We illustrate with toy problems how to use VNCE for both approximate inference and parameter estimation.

An important application of (unnormalised) latent variable models is missing data problems. However, it is not easy to learn from missing data using variational methods with inference networks [55, 54, 38]. We discuss the core problem in detail in Section 5.1.2, and then propose an approximate algorithm, CDI, that offers a novel way to leverage inference networks and applies to both standard variational inference and VNCE.

Finally, We demonstrate how to learn a multivariate truncated Gaussian with incomplete data using the combination of VNCE and CDI. Our work provides a probabilistically-principled approach to an outstanding problem in the graphical models literature [30].

Chapter 4

Variational Noise-Contrastive Estimation

We first derive the objective function for VNCE, which is a variational lower bound to the NCE objective. We then derive theoretical results for VNCE that parallel those for standard variational inference on the log-likelihood. Finally, we illustrate the theory with simple experiments, showing how to perform both approximate inference and parameter estimation with VNCE.

4.1 The method

Gutmann[1] derived a *variational* lower bound to the NCE objective function, which is the mathematical keystone to this thesis. Inspired by the derivation of the variational lower bound for the log-likelihood (2.15), we apply a similar strategy to bounding the NCE objective function in Equation 2.25.

[1] This derivation is unpublished; it was intended as the seed of this thesis.

The intractable integral, $\int \phi(\mathbf{u}, \mathbf{z}; \boldsymbol{\theta})\, \mathrm{d}\mathbf{z}$, appears twice in the NCE objective, both inside the expectation wrt the data

$$\mathbb{E}_{\mathbf{x}} \log \left(\frac{\int \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})\, \mathrm{d}\mathbf{z}}{\int \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})\, \mathrm{d}\mathbf{z} + \nu p_{\mathbf{y}}(\mathbf{x})} \right) \tag{4.1}$$

and in the second term:

$$\nu \mathbb{E}_{\mathbf{y}} \log \left(\frac{\nu p_{\mathbf{y}}(\mathbf{y})}{\int \phi(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta})\, \mathrm{d}\mathbf{z} + \nu p_{\mathbf{y}}(\mathbf{y})} \right). \tag{4.2}$$

Let us focus only on the first term (4.1),

$$\log \left(\frac{\int \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})\, \mathrm{d}\mathbf{z}}{\int \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})\, \mathrm{d}\mathbf{z} + \nu p_{\mathbf{y}}(\mathbf{x})} \right) = -\log \left(1 + \nu \frac{p_{\mathbf{y}}(\mathbf{x})}{\int \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})\, \mathrm{d}\mathbf{z}} \right) \tag{4.3}$$

$$= g(r(\mathbf{x}; \boldsymbol{\theta})). \tag{4.4}$$

where

$$g(r) := -\log\left(1 + \nu \frac{1}{r}\right) : \text{concave function of } r \tag{4.5}$$

and

$$r(\mathbf{x}; \boldsymbol{\theta}) = \frac{\int \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})\, \mathrm{d}\mathbf{z}}{p_{\mathbf{y}}(\mathbf{x})}. \tag{4.6}$$

rewrite r as an expectation

Fact.

$$r(\mathbf{x}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z} \mid \mathbf{x})} \left(\frac{\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})}{q(\mathbf{z} \mid \mathbf{x}) p_{\mathbf{y}}(\mathbf{x})} \right) \tag{4.7}$$

apply Jensen's inequality==>

$$g(r(\mathbf{x}; \boldsymbol{\theta})) \geq \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z} \mid \mathbf{x})} g \left(\frac{\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})}{q(\mathbf{z} \mid \mathbf{x}) p_{\mathbf{y}}(\mathbf{x})} \right) \tag{4.8}$$

(

contrastive variational ineq.

$$= \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z} \mid \mathbf{x})} \log \left(\frac{\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})}{\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) + \nu q(\mathbf{z} \mid \mathbf{x}) p_{\mathbf{y}}(\mathbf{x})} \right), \tag{4.10}$$

It does not appear possible to apply the same combination of importance sampling and Jensen's inequality to the second term of the NCE objective (4.2). However, we don't necessarily need to. The intractable integral in the second term can be approximated just with importance sampling, *re-using* the variational distribution q that we got from the first term. The final objective, which we call the **VNCE objective**:

$$J_{\text{VNCE}}(\boldsymbol{\theta}, q) = \mathbb{E}_{\mathbf{x}} \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z} \mid \mathbf{x})} \log \left(\frac{\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})}{\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) + \nu q(\mathbf{z} \mid \mathbf{x}) p_{\mathbf{y}}(\mathbf{x})} \right) + \nu \mathbb{E}_{\mathbf{y}} \log \left(\frac{\nu p_{\mathbf{y}}(\mathbf{y})}{\nu p_{\mathbf{y}}(\mathbf{y}) + \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z} \mid \mathbf{y})} \left[\frac{\phi(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta})}{q(\mathbf{z} \mid \mathbf{y})} \right]} \right). \tag{4.11}$$

By construction, we have that $J_{\text{NCE}}(\boldsymbol{\theta}) \geq J_{\text{VNCE}}(\boldsymbol{\theta}, q)$ for all q. Shortly, in Section 4.3, we show that the optimal q is the model's posterior $p(\mathbf{z} \mid \mathbf{u}; \boldsymbol{\theta})$. We then prove that optimising the VNCE objective wrt $\boldsymbol{\theta}$ and q is a valid form of parameter estimation and approximate inference.

Ingredients for applying VNCE

Having established the objective function (4.11) for VNCE, it is helpful to explicitly lay out the ingredients required for maximising it. We require:

- A noise distribution over the visible variables $p_{\mathbf{y}}$. This density should ideally be cheap to evaluate, feasible to sample from, and similar to the data distribution.
- A variational distribution over latent variables q. This will typically have its own parameters $\boldsymbol{\alpha}$, apart from the case where we set q to equal the model's posterior over latents. This is only possible for models with tractable posteriors.
- If the variational distribution q has parameters $\boldsymbol{\alpha}$, then we need to compute $\nabla_{\boldsymbol{\alpha}} J_{\text{VNCE}}(\boldsymbol{\alpha})$. As discussed in Section 2.5, both the score function estimator and reparameterisation trick are applicable, depending on the model.
- The gradient $\nabla_{\boldsymbol{\theta}} J_{\text{VNCE}}(\boldsymbol{\theta})$. A derivation of this gradient is given in the Appendix A.2. The key term required to compute it is the gradient of the logarithm of the unnormalised model: $\nabla_{\boldsymbol{\theta}}(\log \phi(\mathbf{u}, \mathbf{z}; \boldsymbol{\theta}))$.

4.2 Role of the noise distribution in VNCE

One immediate concern about the VNCE objective function is the use of importance sampling in the second term

$$\phi(\mathbf{y};\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\,|\,\mathbf{y})}\left[\frac{\phi(\mathbf{y},\mathbf{z};\boldsymbol{\theta})}{q(\mathbf{z}\,|\,\mathbf{y})}\right]. \tag{4.12}$$

Monte Carlo approximations to such expectations can be very high variance when q is far from optimal, as is often the case for high-dimensional problems. In our case, the optimal q can be derived

$$\boxed{\phi(\mathbf{y},\mathbf{z};\boldsymbol{\theta}) = \phi(\mathbf{y};\boldsymbol{\theta})p(\mathbf{z}\,|\,\mathbf{y};\boldsymbol{\theta}),} \tag{4.13}$$

where the conditional distribution is normalised and the factorisation holds because the unnormalised distributions on either side of the equation have the same partition function

$$\int\int \phi(\mathbf{y},\mathbf{z};\boldsymbol{\theta})\,\mathrm{d}\mathbf{z}\,\mathrm{d}\mathbf{y} = \int \phi(\mathbf{y};\boldsymbol{\theta})\,\mathrm{d}\mathbf{y}. \tag{4.14}$$

==>

$$\phi(\mathbf{y};\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\,|\,\mathbf{y})}\left[\frac{\phi(\mathbf{y},\mathbf{z};\boldsymbol{\theta})}{q(\mathbf{z}\,|\,\mathbf{y})}\right] \tag{4.15}$$

$$= \phi(\mathbf{y};\boldsymbol{\theta})\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\,|\,\mathbf{y})}\left[\frac{p(\mathbf{z}\,|\,\mathbf{y};\boldsymbol{\theta})}{q(\mathbf{z}\,|\,\mathbf{y})}\right]. \tag{4.16}$$

Hence, the variance of a Monte Carlo estimate of the expectation in Equation 4.15 will equal the variance of a Monte Carlo estimate of the expectation in Equation 4.16. When $q(\mathbf{z}\,|\,\mathbf{y}) = p(\mathbf{z}\,|\,\mathbf{y};\boldsymbol{\theta})$, the latter expectation equals one, yielding a zero-variance Monte Carlo estimate.

This result is encouraging, since VNCE optimises $q(\mathbf{z}\,|\,\mathbf{y})$ to approximate $p(\mathbf{z}\,|\,\mathbf{y};\boldsymbol{\theta})$, as we prove in Section 4.3. *However*, q only receives gradient signal from the first term of the objective, which is evaluated using data samples. We do not fit our approximate posterior using noise samples. If the noise is sufficiently different from the data, then this may be problematic, since q will be a poor fit for the true posterior in regions where the noise, but not data, is likely. This will increase the variance of the importance sampled estimates.

Whilst the issue of importance sampling deserves empirical investigation, the preceding analysis suggests that it can be managed by choosing a good noise distribution. Since NCE relies on a good choice of noise anyway, we have not introduced any new problems. That said, the problem of choosing a good noise, one that resembles the data, may become harder for latent variable models. Often the underlying premise of latent variable modelling is that the data distribution is complex; we need to use an intractable average of many conditional distributions (conditioning on latent states) to properly describe it. It is not clear how to choose the noise in this scenario, although work such as that of Ceylan and Gutmann [11], which semi-automates the choice of noise, appear promising. Finally, we note that choosing a noise may be less difficult in the case of small amounts of missing data, since fitting a noise to the observed data is still possible.

4.3 Theoretical analysis of VNCE

We here prove that VNCE has the same maximum as NCE, which is obtained when the variational distribution q equals the model's posterior over latents $p(\mathbf{z} \mid \mathbf{u}; \boldsymbol{\theta})$. This is the same optimal q as in standard variational inference, and implies that VNCE is not only useful for parameter estimation, but also performs approximate inference of the latent variables.

Standard variational inference minimises the KL-divergence between the approximate and true posterior. In contrast, we show that VNCE minimises a different *f-divergence* between the two posteriors. This is likely to have important consequences on the type of approximate posterior we learn whenever q is not flexible enough to match the true posterior. We discuss these consequences in more detail after deriving the necessary equations.

Definition 4.3.1. An *f-divergence* $D_f(p \parallel q)$ between two probability densities p and q, is defined as

$$D_f(p \parallel q) = \mathbb{E}_{u \sim q}\left[f\left(\frac{p(u)}{q(u)} \right) \right], \tag{4.17}$$

where f is a convex function satisfying $f(1) = 0$.

These two properties make f-divergences useful for learning a data generating distribution p by optimising q to minimise $D_f(p(u) \| q(u))$. In fact, ML learning minimises the KL divergence, which is one example of an f-divergence where $f(u) = u \log(u)$.

Lemma 1. contrastive variational identity:

$$J_{\text{NCE}}(\boldsymbol{\theta}) - J_{\text{VNCE}}(\boldsymbol{\theta}, q) = \mathbb{E}_{\mathbf{x}}\left[D_{f_{\mathbf{x}}}(p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x}) \parallel q(\mathbf{z} \mid \mathbf{x}))\right], \tag{4.20}$$

where

$$f_{\mathbf{x}}(u) = \log(v_{\mathbf{x}} + (1 - v_{\mathbf{x}})u^{-1}), \qquad v_{\mathbf{x}} = \frac{\phi_{\boldsymbol{\theta}}(\mathbf{x})}{\phi_{\boldsymbol{\theta}}(\mathbf{x}) + \nu p_{\mathbf{y}}(\mathbf{x})}. \tag{4.21}$$

Moreover, this f-divergence can be expressed as the difference of two KL-divergences

$$D_{f_{\mathbf{x}}}(p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x}) \parallel q(\mathbf{z} \mid \mathbf{x})) = D_{KL}(q(\mathbf{z} \mid \mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x})) - D_{KL}(q(\mathbf{z} \mid \mathbf{x}) \parallel m_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{x})), \tag{4.22}$$

where $m_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{x}) = v_{\mathbf{x}} p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x}) + (1 - v_{\mathbf{x}}) q(\mathbf{z} \mid \mathbf{x})$ *is a convex combination of the true and approximate posteriors.*

Proof. Key to this proof is the previously discussed factorisation of the model,

$$\boxed{\phi_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = \phi_{\boldsymbol{\theta}}(\mathbf{x}) p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x}),} \tag{4.23}$$

[2] Technically, q and p need only be equal 'a.e.' wrt the Lebesgue measure. See Bartle [3] for more details.

[3] Throughout the following derivations, parameters are moved into the subscript for compactness.

where the conditional distribution is normalised. With this factorisation at hand, we now consider the difference between the NCE objective: $J_{\text{NCE}}(\boldsymbol{\theta})$ (2.25) and the VNCE objective: $J_{\text{VNCE}}(\boldsymbol{\theta}, q)$ (4.11). Each objective consists of two terms: the first is a nexpectation wrt the data, the second a nexpectation wrt the noise distribution $p_{\mathbf{y}}$. The second terms of J_{NCE} and J_{VNCE} are identical, so their difference equals the difference between their first terms

contrastive log-likelihood — contrastive variational lower bound

$$
\begin{aligned}
&= \log\left(\frac{\phi_{\boldsymbol{\theta}}(\mathbf{x})}{\phi_{\boldsymbol{\theta}}(\mathbf{x}) + \nu p_{\mathbf{y}}(\mathbf{x})}\right) - \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\mid x)} \log\left(\frac{\phi_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}{\phi_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z}) + \nu p_{\mathbf{y}}(\mathbf{x}) q(\mathbf{z}\mid\mathbf{x})}\right) && (4.24)\\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\mid x)}\left[\log\left(\frac{\phi_{\boldsymbol{\theta}}(\mathbf{x})}{\phi_{\boldsymbol{\theta}}(\mathbf{x}) + \nu p_{\mathbf{y}}(\mathbf{x})}\right) + \log\left(1 + \frac{\nu p_{\mathbf{y}}(\mathbf{x}) q(\mathbf{z}\mid\mathbf{x})}{\phi_{\boldsymbol{\theta}}(\mathbf{x}) p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x})}\right)\right] && (4.25)\\
&\qquad \left(\text{———————}\right.\\
&\qquad \left(\text{———————}\right.\\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\mid x)}\left[\log\left(\frac{\phi_{\boldsymbol{\theta}}(\mathbf{x})}{\phi_{\boldsymbol{\theta}}(\mathbf{x}) + \nu p_{\mathbf{y}}(\mathbf{x})} + \left(1 - \frac{\phi_{\boldsymbol{\theta}}(\mathbf{x})}{\phi_{\boldsymbol{\theta}}(\mathbf{x}) + \nu p_{\mathbf{y}}(\mathbf{x})}\right)\frac{q(\mathbf{z}\mid\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x})}\right)\right] && (4.28)\\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\mid x)}\left[\log\left(v_{\mathbf{x}} + (1 - v_{\mathbf{x}})\frac{q(\mathbf{z}\mid\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x})}\right)\right] && (4.29)\\
&= \left[D_{f_{\mathbf{x}}}(p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x}) \parallel q(\mathbf{z}\mid\mathbf{x}))\right], && (4.30)
\end{aligned}
$$

where $f_{\mathbf{x}}(u) = \log(v_{\mathbf{x}} + (1 - v_{\mathbf{x}})u^{-1})$. $D_{f_{\mathbf{x}}}$ is a validf-divergence /

We now prove that this f-divergence can be expressed as the difference of two KL-divergences as in Equation 4.22. To do this, we pull q/p outside of the log in Equation 4.29,

$$
\begin{aligned}
&D_{f_{\mathbf{x}}}(p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x}) \parallel q(\mathbf{z}\mid\mathbf{x}))\\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\mid\mathbf{x})}\left[\log\frac{q(\mathbf{z}\mid\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x})}\right] + \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}\mid\mathbf{x})}\left[\log\left(v_{\mathbf{x}}\frac{p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x})}{q(\mathbf{z}\mid\mathbf{x})} + (1 - v_{\mathbf{x}})\right)\right] && (4.31)\\
&= D_{KL}(q(\mathbf{z}\mid\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x})) - D_{KL}(q(\mathbf{z}\mid\mathbf{x}) \parallel m_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})).
\end{aligned}
$$

where $m_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x}) = v_{\mathbf{x}} p_{\boldsymbol{\theta}}(\mathbf{z}\mid\mathbf{x}) + (1 - v_{\mathbf{x}})q(\mathbf{z}\mid\mathbf{x})$.

(4.33)

□

Lemma 1 tells us that the optimal q is the model's posterior. This follows from a key property of the f-divergence stated in Equation 4.19. Hence, VNCE has the same optimum as standard variational inference. However, in VNCE we don't just minimise $D_{KL}(q(\mathbf{z} \mid \mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x}))$; we minimise the difference between it and $D_{KL}(q(\mathbf{z} \mid \mathbf{x}) \parallel m_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{x}))$. The following corollary gives us some idea of the impact of this extra term, by telling us when it vanishes.

Corollary 1.1. *As* $v_{\mathbf{x}} = \phi_{\boldsymbol{\theta}}(\mathbf{x})/(\phi_{\boldsymbol{\theta}}(\mathbf{x}) + \nu p_{\mathbf{y}}(\mathbf{x})) \to 0$*, our f-divergence tends to the standard KL-divergence,*

$$D_{f_{\mathbf{x}}}(p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x}) \parallel q(\mathbf{z} \mid \mathbf{x})) \;\to\; D_{KL}(q(\mathbf{z} \mid \mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x})). \tag{4.34}$$

In particular, ***as the ratio of noise to data,*** $\nu \to \infty$***, we recover the KL-divergence from standard variational inference.***

The fact that we recover the standard KL-divergence as $\nu \to \infty$ agrees with a theoretical result for NCE, which states that as $\nu \to \infty$, NCE == ML.[4]

Lemma 1 tells us that VNCE transforms *inference* in unnormalised models into an optimisation problem. The end result of this optimisation is an approximate posterior that can often be directly useful, independent of the full model ϕ. This is the case in many semi-supervised learning tasks for instance [26]. Of course, approximating the model's posterior is only useful if we have learned good parameters $\boldsymbol{\theta}$ for the model

The following theorem provides some assurance of this: it shows that the maximum of the VNCE objective matches the maximum of the NCE objective.

Theorem 2. *(Equivalence of VNCE and NCE)*

$$\max_{\boldsymbol{\theta}} J_{\text{NCE}}(\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \max_{q} J_{\text{VNCE}}(\boldsymbol{\theta}, q) \tag{4.36}$$

[4] 'Equivalent' in the sense that the NCE objective converges, under mild conditions, to a Poisson-transformed likelihood function [2]. This then implies that NCE has the same asymptotic properties as maximum likelihood estimation.

Corollary 2.1. *(EM algorithm for VNCE) Given a random initial starting point* $\boldsymbol{\theta}_0$*, and the following optimisation procedure:*

1. *(E-step)* $q_k(\mathbf{z} \mid \mathbf{u}) = p(\mathbf{z} \mid \mathbf{u}; \boldsymbol{\theta}_k)$ max θ
2. *(M-step)* $\boldsymbol{\theta}_{k+1} = \arg\max_{\boldsymbol{\theta}} J_{\text{VNCE}}(\boldsymbol{\theta}, q_k)$
3. *Unless converged, repeat steps 1 and 2*

==>*, for all* $k \in \mathbb{N}$*, we have:* $J_{\text{NCE}}(\boldsymbol{\theta}_{k+1}) \geq J_{\text{NCE}}(\boldsymbol{\theta}_k)$.

Proof. Let $k \in \mathbb{N}$. After the E-step of optimisation, we have $q_k(\mathbf{z} \mid \mathbf{u}) = p(\mathbf{z} \mid \mathbf{u}; \boldsymbol{\theta}_k)$ and so, by Lemma 1,

$$J_{\text{NCE}}(\boldsymbol{\theta}_k) - J_{\text{VNCE}}(\boldsymbol{\theta}_k, q_k) = \mathbb{E}_{\mathbf{x}}\left[D_{f_{\mathbf{x}}}(p(\mathbf{z} \mid \mathbf{x}; \boldsymbol{\theta}_k) \parallel p(\mathbf{z} \mid \mathbf{x}; \boldsymbol{\theta}_k))\right] = 0, \tag{4.40}$$

implying that $J_{\text{VNCE}}(\boldsymbol{\theta}_k, q_k) = J_{\text{NCE}}(\boldsymbol{\theta}_k)$. Now, in the M-step of optimisation, we have

$$\boldsymbol{\theta}_{k+1} = \arg\max_{\boldsymbol{\theta}} J_{\text{VNCE}}(\boldsymbol{\theta}, q_k) \implies J_{\text{VNCE}}(\boldsymbol{\theta}_{k+1}, q_k) \geq J_{\text{VNCE}}(\boldsymbol{\theta}_k, q_k)\ , \tag{4.41}$$

finally, by using Lemma 1 again, we see that $J_{\text{NCE}}(\boldsymbol{\theta}_{k+1}) \geq J_{\text{VNCE}}(\boldsymbol{\theta}_{k+1}, q_k)$. Putting everything together,

$$J_{\text{NCE}}(\boldsymbol{\theta}_{k+1}) \geq J_{\text{VNCE}}(\boldsymbol{\theta}_{k+1}, q_k) \geq J_{\text{VNCE}}(\boldsymbol{\theta}_k, q_k) = J_{\text{NCE}}(\boldsymbol{\theta}_k)\ . \tag{4.42}$$

□

As is the case for standard EM, the above result does not hold if we only take a 'partial' E-step, by making q close, but not exactly equal, to $p(\mathbf{z} \mid \mathbf{x}; \boldsymbol{\theta})$. Thus, any approach using a non-exact, variational q will not have such strong theoretical guarantees. However, the corollary still holds if we take a partial M-step, increasing the value of $J_{\text{VNCE}}(\boldsymbol{\theta}, q_k)$ through a few gradient steps without computing the true argmax. GEM for VNCE

4.4 Toy model illustrating approximate inference with VNCE

The following toy model illustrates Lemma 1, which described the divergence minimised when using VNCE for approximate inference. The model is normalised and has 2-dimensional latents and visibles,

$$\mathbf{z} \sim \mathcal{N}(0, \mathbb{I}), \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, c\mathbb{I}), \quad \boldsymbol{\zeta}_\mathbf{z} = \begin{bmatrix} z_1 z_2 \\ z_1 z_2 \end{bmatrix}, \tag{4.43}$$

$$\mathbf{x} = \boldsymbol{\zeta}_\mathbf{z} + \boldsymbol{\epsilon}, \tag{4.44}$$

so that,

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, \mathbb{I}), \qquad p(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\zeta}_\mathbf{z}, c^2\mathbb{I}), \qquad p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x} \mid \mathbf{z}), \tag{4.45}$$

where c is a known standard deviation that we set to 0.3. Because c is known, this model has no parameters for us to estimate; we are solely interested in approximating the posterior distribution $p(\mathbf{z} \mid \mathbf{x})$, given by

$$p(\mathbf{z} \mid \mathbf{x}) = \frac{\mathcal{N}(\mathbf{z}; 0, \mathbb{I})\ \mathcal{N}(\mathbf{x}; \boldsymbol{\zeta}_\mathbf{z}, c\mathbb{I})}{\int \mathcal{N}(\mathbf{z}; 0, \mathbb{I})\ \mathcal{N}(\mathbf{x}; \boldsymbol{\zeta}_\mathbf{z}, c\mathbb{I})\, \mathrm{d}\mathbf{z}}. \tag{4.46}$$

This integral in the denominator is not easy, perhaps impossible, to solve in closed form. We therefore approximate $p(\mathbf{z}|\mathbf{x})$ with a distribution $q(\mathbf{z}|\mathbf{x}; \boldsymbol{\alpha}) = \mathcal{N}(\mathbf{z};\ \boldsymbol{\mu}(\mathbf{x}; \boldsymbol{\alpha}), \boldsymbol{\Sigma}(\mathbf{x}; \boldsymbol{\alpha}))$, where $\boldsymbol{\Sigma}$ is a diagonal covariance matrix and $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are parametrised by a single 2-layer feed-forward neural network. The details of this parametrisation, and the training of the neural network, are given in the Appendix (A.3).

Given this q, we maximise the VNCE objective function (4.11) with respect to $\boldsymbol{\alpha}$. This optimisation depends on the choice of noise distribution $p_\mathbf{y}$ and the ratio of noise-to-data samples ν. We consider two choices for the noise,

$$p^1_\mathbf{y}(\mathbf{u}) = \mathcal{N}(\mathbf{u};\ \bar{\mathbf{x}}, \bar{\Sigma}), \qquad p^2_\mathbf{y}(\mathbf{u}) = \mathcal{N}(\mathbf{u};\ 0, 10\mathbb{I}), \tag{4.47}$$

where $\bar{\mathbf{x}}$ and $\bar{\Sigma}$ are the empirical mean and covariance, respectively. Figure 4.1 compares these noise distributions to the data-generating distribution $p(\mathbf{x})$. For each noise, we train with $\nu = 1$, and then for the second noise distribution we also train with $\nu = 100$. The approximate posteriors learnt with these settings are shown in the last three rows

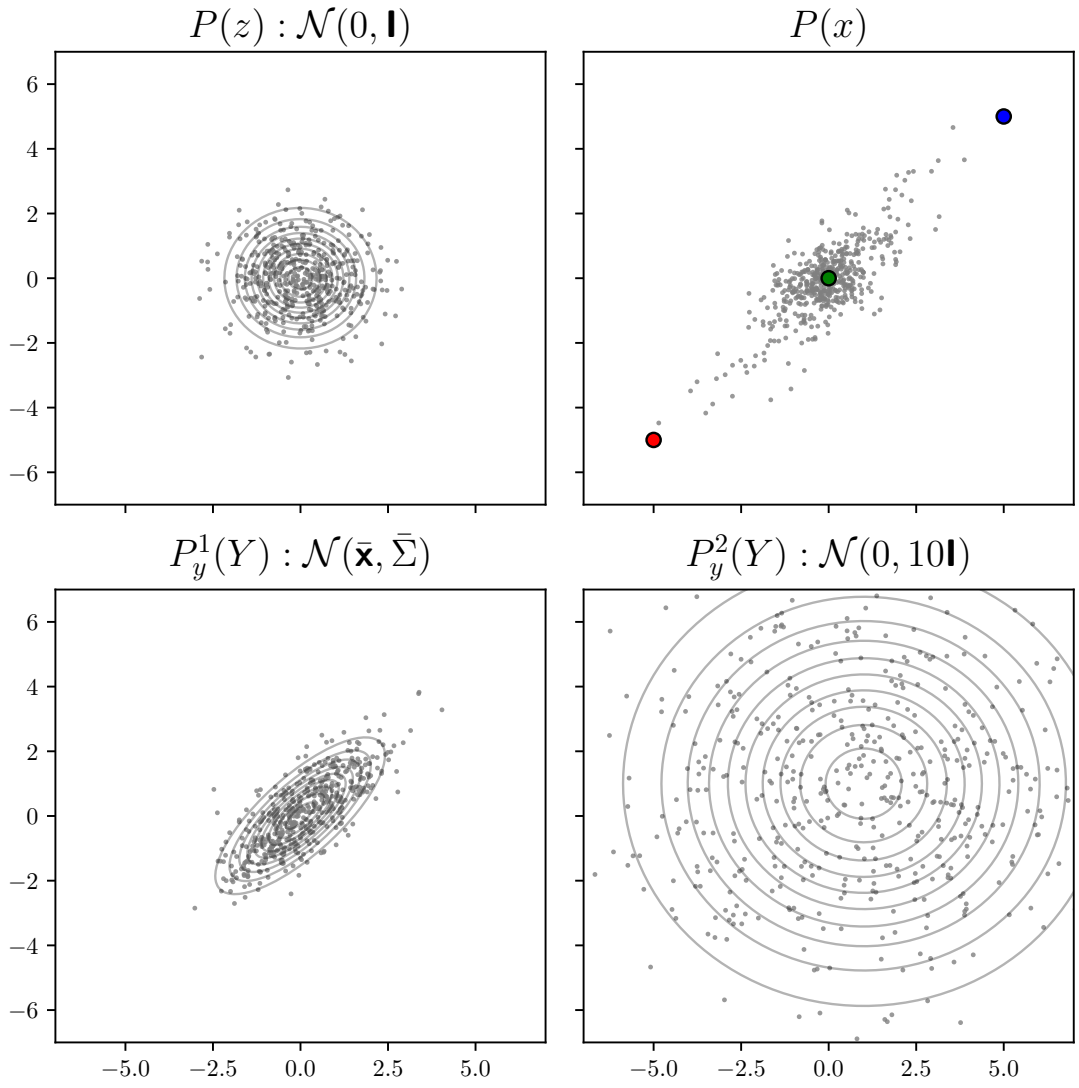

Fig. 4.1 **Top row**. The marginals of a simple latent variable model defined in Equations 4.43 and 4.45. The red, green and blue points in the rightmost plot are three 'landmark' points for which we approximate the posterior $p(\mathbf{z} \mid \mathbf{x})$—see Figure 4.2 for the results. **Bottom row**. Two choices of the noise distribution used in VNCE when approximating $p(\mathbf{z} \mid \mathbf{x})$. The leftmost noise was chosen by fitting a Gaussian to samples from $p(\mathbf{x})$. It has a similar mean and covariance structure to $p(\mathbf{x})$, so the two are well-matched at the mode (green landmark). However, it has much lighter tails, and is therefore a bad match to $p(\mathbf{x})$ at the red and blue landmarks. The rightmost noise represents a 'bad' choice of noise that is not a particularly good match to the true data generating distribution anywhere.

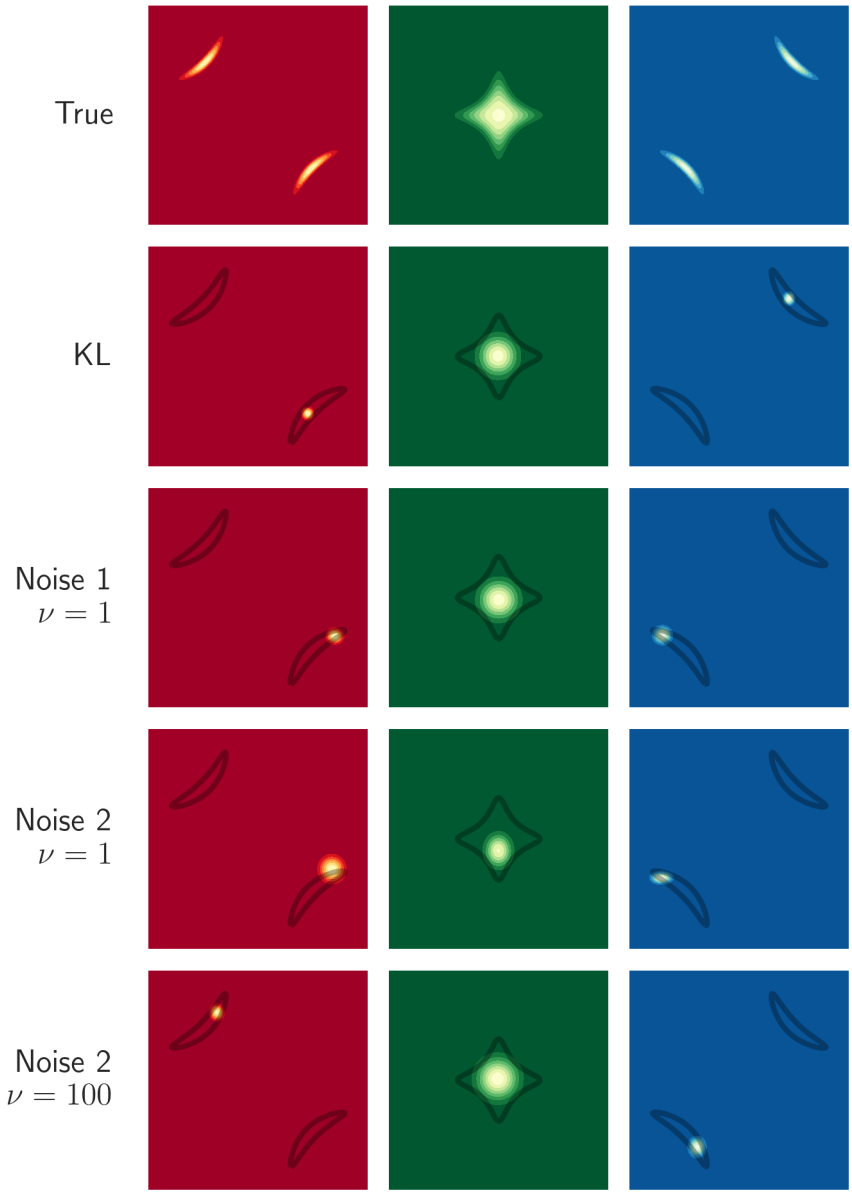

Fig. 4.2 Density plots of true and approximate posteriors for the 2D toy model defined in Equations 4.43 and 4.45. The colour-coded columns correspond to the landmark $\mathbf{x}$ points in the top-right plot of Figure 4.1, which we condition on when computing posteriors. **Top row**. The true posteriors, $p(\mathbf{z} \mid \mathbf{x})$. Depending on the $\mathbf{x}$ we are conditioning on, the posteriors can have very different shapes. **Second row**. The approximate posteriors we obtain when applying standard variational inference, minimising the KL-divergence. **Third row**. The approximate posterior learned when applying VNCE with the first choice of noise distribution—see bottom-left plot of Figure 4.1. **Final two rows**. Approximate posteriors learned with VNCE, using the second noise—see bottom-right plot of Figure 4.1. The final row increases the ratio of noise to data, making VNCE behave more like standard variational inference.

of Figure 4.2. The first two rows show the true posterior, calculated with numerical integration, and the approximate posterior learnt by maximising the ELBO (2.15) and thereby minimising the KL divergence between $q(\mathbf{z}|\mathbf{x};\boldsymbol{\alpha})$ and $p(\mathbf{z}|\mathbf{x})$.

The ground-truth row in Figure 4.2 illustrates how a relatively simple model can generate complex posteriors that vary greatly depending on which $\mathbf{x}$ we condition on. For some $\mathbf{x}$, the model generates relatively well-behaved, uni-modal posteriors resembling a Laplacian (green), whilst for other $\mathbf{x}$, the model generates multi-modal posteriors concentrated along curved manifolds (blue & red). Fitting an approximate posterior to such diverse distributions is challenging, which is why a highly expressive function approximator such as a neural network is needed. Even with a neural network, our current parametrisation restricts us to outputting a Gaussian with diagonal covariance, which is a poor match for any of the true posteriors.

We see that the approximate posterior learned by minimising the KL-divergence is 'mode-seeking', latching onto a single density peak, whilst *underestimating* the variance of this peak. This behaviour is well-known and predictable from the mathematical form of the KL in Equation 2.16, since if q places any probability mass in a region where p is unlikely, then the denominator in the log is close to zero, exploding the KL.

Of greater interest to us is the fact that approximate posteriors learnt with VNCE display similar low-variance, mode-seeking behaviour. The key difference with VNCE is that the choice of noise distribution and ratio of noise-to-data samples, ν, ought to affect the shape of the posteriors. In particular, Lemma 1 and associated Corollary 1.1 tell us that VNCE minimises an f-divergence which tends toward the KL when $\phi_{\boldsymbol{\theta}}(\mathbf{x})/(\phi_{\boldsymbol{\theta}}(\mathbf{x}) + \nu p_{\mathbf{y}}(\mathbf{x})) \to 0$. That is, when the noise assigns a high density to $\mathbf{x}$ compared to the model, or when ν is large, we expect VNCE to produce a similar posterior to the ELBO at that *specific* point $\mathbf{x}$. In contrast, as this ratio $\to 1$, our f-divergence should become a worse approximation to the KL. This should occur for those $\mathbf{x}$ points which are unlikely under the noise, compared to the model.

We see that this theory is borne out in practice. Noise 1 assigns a high density to the green landmark, and so the corresponding posterior resembles that obtained using the KL—see row 3 of Figure 4.2. The same is true of noise 2 when $\nu = 100$, but *not* when $\nu = 1$, as can be seen in rows 4 and 5 of Figure 4.2. This is because noise 2 is very diffuse and assigns a low density to the green landmark compared to the model. All three setting of VNCE produce posteriors that are slightly different to the KL at the extreme red and blue landmarks. This is because neither noise is a good match to the model here, and whilst increasing ν helps, it does not entirely fix the issue.

To summarise, even though the VNCE posteriors do not always match the KL posteriors, when the noise is good and/or when ν is large, we obtain reasonable approximations to the true posteriors (modulo the restrictive parametric assumptions).

4.5 Toy model illustrating parameter estimation with VNCE

We consider a simple mixture of two Gaussians model, where the only unknown parameter is the standard deviation of one of the Gaussians. We consider two cases: firstly a normalised MoG, and then an unnormalised version. Finally, for the unnormalised version, we run a population analysis to compare the performance of VNCE against both NCE and MLE across multiple sample sizes.

4.5.1 Normalised mixture of Gaussians

A common latent variable model is the mixture of Gaussian (MoG) model For a mixture with two components, a single binary latent variable $z \sim \mathcal{B}er(\pi)$ determines which of two Gaussians the data was generated from. If $\pi = 1/2$, we have the following joint pdf, $p(u, z; \theta)$,

$$p(u, z; \theta) = \frac{(1-z)}{2}\mathcal{N}(u; 0, \theta^2) + \frac{z}{2}\mathcal{N}(u; 0, \sigma_1^2). \tag{4.48}$$

We assume that the variance of the second component, σ_1^2, is known, and our task is to estimate the value of θ. Whilst this estimation problem does not involve an unnormalised model, we can still apply VNCE to it. This provides a useful sanity check of the method's validity in a simple setting where we do not have a scaling parameter to estimate in conjunction with θ.

For a simple experiment, we set $\sigma_1 = 1$ and let $\theta^* = 4$ be the true value of θ. We then make the following choices of noise and variational distribution,

$$p_{\mathbf{y}}(u) = \mathcal{N}(u; 0, \theta^{*2}) \tag{4.49}$$

$$q(z|u) = p(z = 0 \mid u; \theta) = \left(1 + \frac{\theta_k}{\sigma_1}\exp\left(\frac{-u^2}{2}\left(\frac{1}{\sigma_1^2} - \frac{1}{\theta^2}\right)\right)\right)^{-1}. \tag{4.50}$$

[5] The model was first normalised before applying MLE.

The choice of noise distribution is not too important for this simple model, so long as it approximately matches the data generating distribution. With this setup, we can then apply the EM type algorithm presented in Corollary 2.1. Figure 4.3 illustrates the results with plots of the NCE and VNCE objectives obtained after each E-step and M-step during learning. It is clear from the figure that the value of the NCE objective at the current parameter (red-dashed line) never decreases, in accordance with Corollary 2.1. Moreover, the figure validates Theorem 2, which states that the maximum of the VNCE objective with respect to θ and q equals the maximum of the NCE objective with respect to θ. We see this from the overlap of the blue circle (maximum of NCE) and the red square (maximum of VNCE) in the final plot (bottom-right).

4.5.2 Unnormalised mixture of Gaussians

We would like to modify the MoG given in Equation 4.48 to obtain an unnormalised family. An obvious candidate is

$$\phi(u, z; \theta, c) = e^{-c}\left((1-z)e^{-\frac{u^2}{2\theta^2}} + ze^{-\frac{u^2}{2\sigma_1^2}}\right) \tag{4.51}$$

where e^{-c} is a scaling parameter needed for NCE, as discussed at the end of Section 2.6. It is important to note that the usual MoG family in Equation 4.48 is *not* nested within unnormalised family in Equation 4.51. We can see this more easily by normalising 4.51, giving us

$$p(u, z, \theta) = (1-z)(1-\pi_\theta)\mathcal{N}(u; 0, \theta) + z\pi_\theta\mathcal{N}(u; 0, \sigma_1), \tag{4.52}$$

where $\pi_\theta = \sigma_1/(\theta+\sigma_1)$. We see that the parameter θ of the unnormalised MoG controls both the width of the first component and the mixture probability π_θ.

Equation 4.52 tells us how to simulate data from our unnormalised model ϕ. First sample $z \sim \mathcal{B}er(\frac{\sigma_1}{\theta+\sigma_1})$ and then sample $\{x_1, ..., x_i\}$ data points

$$x_i \sim \left\{ \begin{array}{ll} \mathcal{N}(u; 0, \theta^2), & \text{if } z = 0 \\ \mathcal{N}(u; 0, \sigma_1^2), & \text{if } z = 1 \end{array} \right\}. \tag{4.53}$$

For our experiments, we simulate observed data with $\sigma_1 = 1$ and $\theta^* = 4$. Using this synthetic data, we can proceed as before, using an EM algorithm to learn the parameter

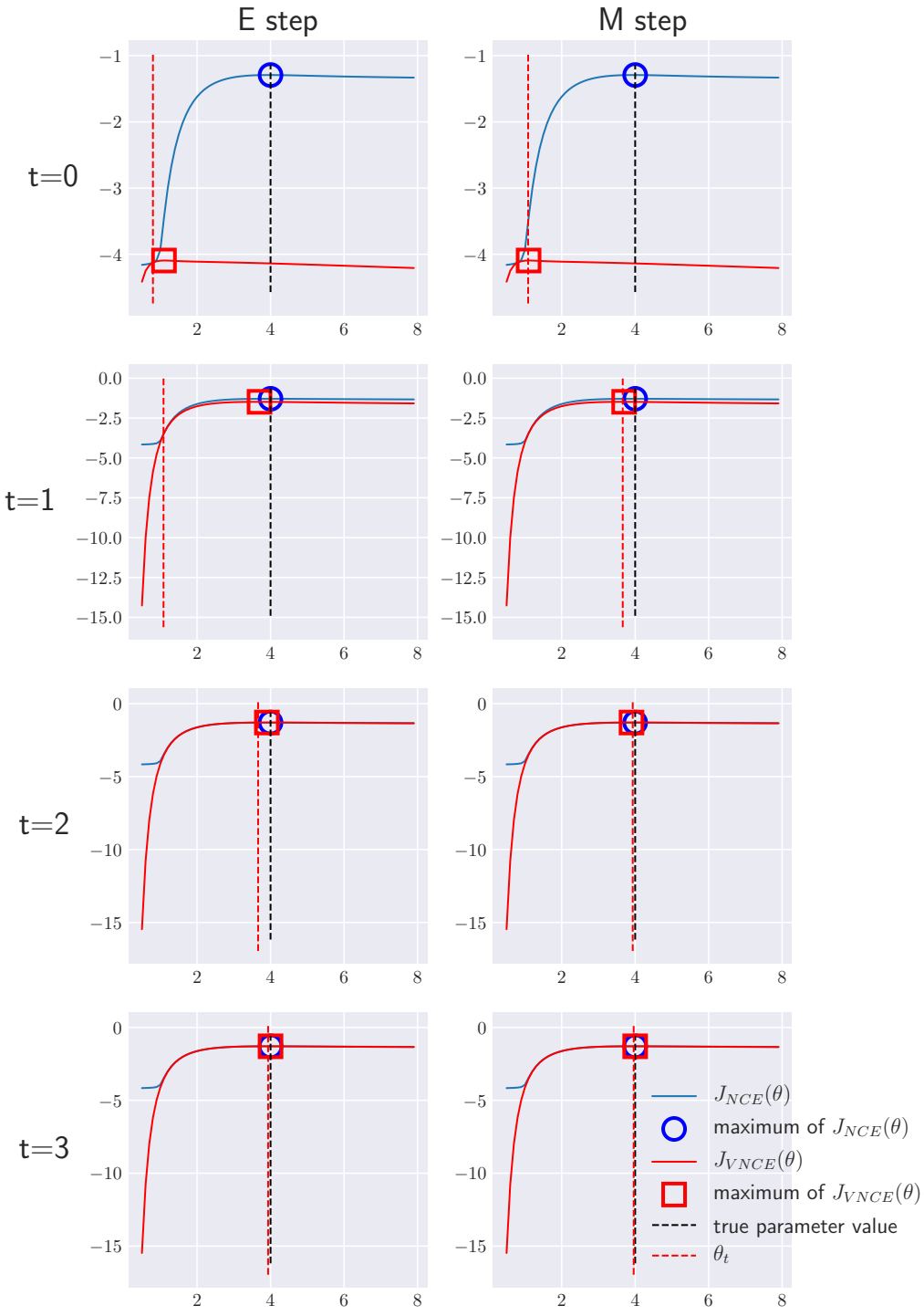

Fig. 4.3 EM-type algorithm for VNCE applied to a normalised MoG model. The figure reads row-by-row, from left to right. The blue solid line is the NCE objective, with a blue circle at its maximum. The red solid line is the VNCE objective for our current vlb $q(z|u)$, with a red square at its maximum. The black dashed line is the ground truth parameter value, whilst the red dashed line is our current estimate of the model's parameter. In the E-step, we set $q(z|u)$ = the true posterior $p(z|u;\theta_k)$. In the M-step we optimise the model's parameter using the VNCE objective, and hence the red dashed line shifts to the centre of the red square.

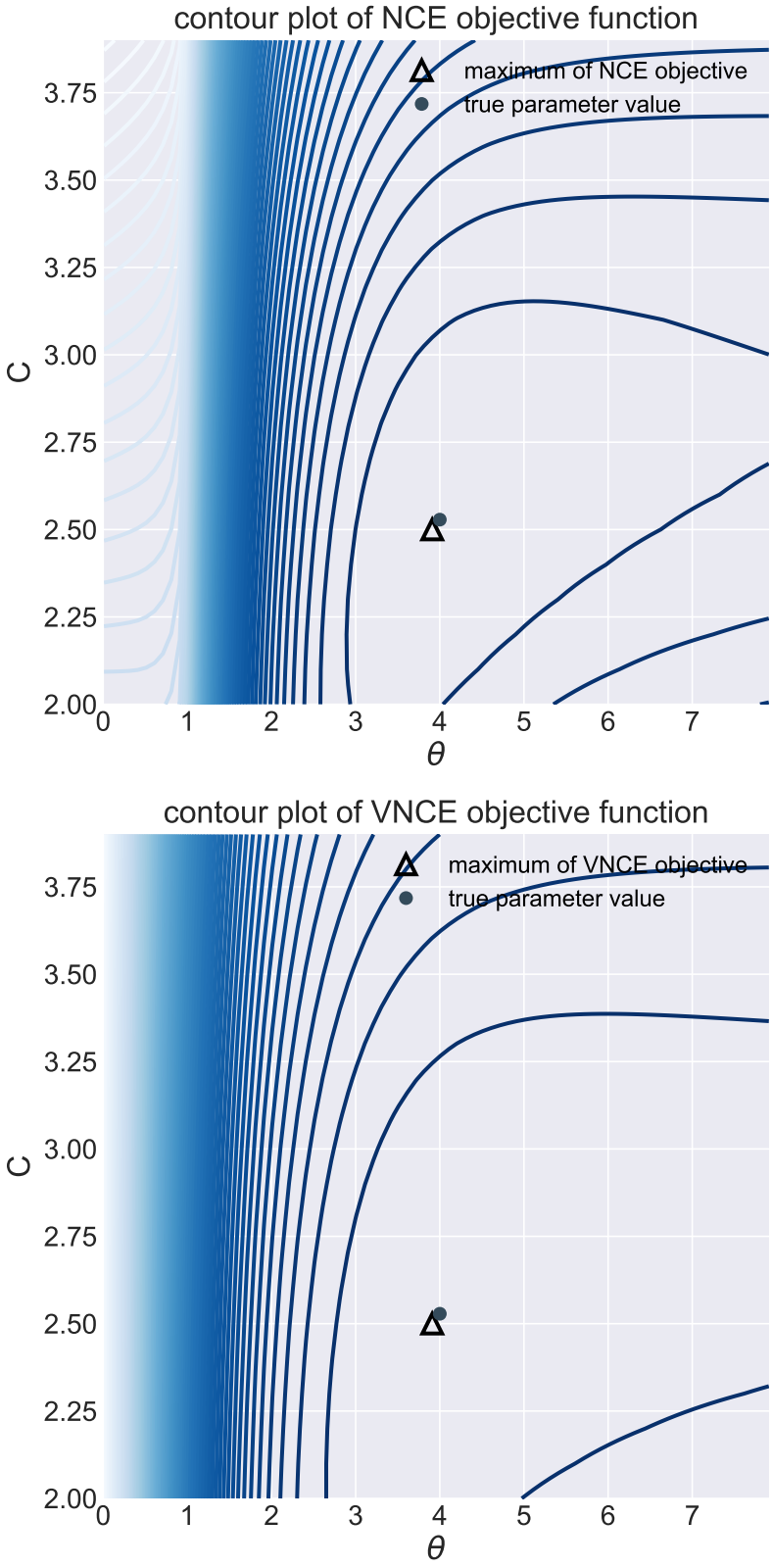

Fig. 4.4 Contour plot of true NCE objective function (top) compared to the VNCE objective function (bottom) for an unnormalised mixture of Gaussians. For the VNCE objective we have set the variational distribution $q(z|u)$ to be the posterior $p(z|u;\tilde{\theta})$, where $\tilde{\theta}$ is our final estimate obtained after optimisation. The vertical axis corresponds to c, the (log) scaling parameter.

θ. To do this, we would need to use the posterior over latents, given by,

$$p(z = 0 \mid u) = \frac{1}{1 + \exp(\frac{-u^2}{2}(\frac{1}{\sigma_1^2} + \frac{1}{\theta^{*2}}))}. \tag{4.54}$$

However, for more complex models, we do not have access to such a posterior, and so it is important that VNCE still works when we approximate it with a parametrised variational distribution q. To test this, we use the following q,

$$q(z = 0 \mid u; \boldsymbol{w}) = \frac{1}{1 + \exp(w_0 + w_1 u + w_2 u^2)}, \tag{4.55}$$

where $\boldsymbol{w} = (w_0, w_1, w_2)^T$ are variational parameters. This q family clearly contains the true posterior.

Visualising learning is less straightforward now that our parameter set is 2-dimensional, but we can compare the contour plot of the true NCE objective function against the lower bound obtained at the end of learning. Figure 4.4 shows the results. We see that both NCE and VNCE converges to the same correct value of θ, and correct normalising constant.

4.5.3 Population analysis of unnormalised mixture of Gaussians

Figure 4.5 shows the MSE $\mathbb{E}||\theta - \theta^*||^2$ for VNCE, NCE and maximum likelihood across multiple runs and with different sample sizes. To produce it, we generated 500 distinct ground-truth values for standard deviation parameter in the unnormalised MoG, sampling uniformly from the interval $[2, 6]$. For each of the 500 θ^*s, we estimate it using all three estimation methods and with a range of sample sizes. Every run was initialised from five random values and the best result out of the five was kept in order to avoid local optima, which exist since both the likelihood and NCE objective functions are bi-modal.

Figure 4.5 (a) demonstrates that the estimation accuracy of VNCE increases with sample size, and is comparable to that of NCE. This gives some evidence of the consistency of VNCE. We note that the comparison is complicated by the fact that NCE was particularly prone to falling into local optima (despite our use of multiple random initialisations). This is why NCE's worst-case accuracy is significantly worse than that of the other two methods, as shown by the upper dashed blue line.

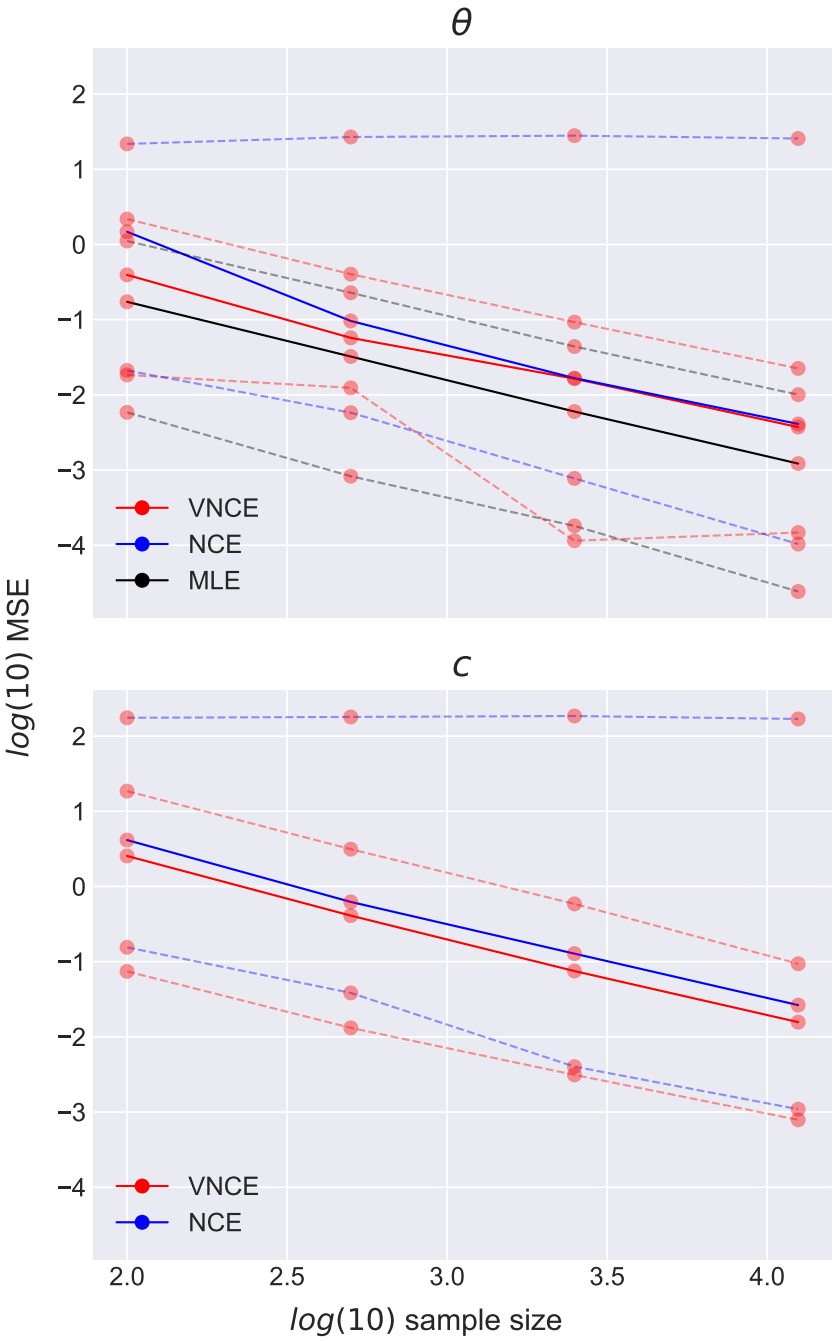

Fig. 4.5 Log sample size against log mean-squared error (MSE) produced when estimating the standard deviation and scaling parameter for 500 different unnormalised mixture of Gaussian models. The thick central lines show the median MSE of the 500 runs, whilst the dashed lines mark the 1st and 9th deciles. **Top**. The MSEs of the model parameter θ. The constant negative slope of the red line is evidence of the consistency of VNCE. **Bottom**. The MSEs of the log-scaling parameter c. The constant negative slop indicates that VNCE estimates the normalisation constant of the model more accurately with larger sample sizes.

Chapter 5

Learning graphical models with incomplete data

This chapter demonstrates how to perform parameter estimation and graph learning for an unnormalised graphical model in the context of missing data. We show that VNCE outperforms a standard baseline method, where data is filled in at the start of learning and treated as fully observed. Our work here offers a potential solution to a recent problem in the graphical models literature [30].

Applying variational approaches to missing data problems is not straightforward [55]. We need to compute all possible conditional distributions over every subset of the variables, which easily becomes intractable. To solve this problem, we introduce a novel variational learning approach called Cumulative Data Imputation (CDI) that only requires univariate conditional distributions. We demonstrate empirically that CDI enables successful learning.

5.1 VNCE for missing data

5.1.1 Missing data mechanisms

The mechanisms that generate missing data can affect the inferences we make about the data-generating process. A clear illustration of this point is found in the work of mathematician Abraham Wald who, during World War II, studied the distribution of damage to aircrafts returning from battle[33]. The returning aircrafts had a highly non-uniform distribution of bullet holes, leading some to believe that the most damaged regions were being explicitly targeted and should therefore be reinforced in subsequent

missions. Wald noted that the *least* damaged regions may well be the weakest, such that any planes damaged there did not return from battle (and therefore represent missing data). In essence, Wald proposed a *model* of how the missingness of a datapoint (aircraft) is conditional on its features (where it was shot and the strength of its components).

This idea of modelling the missing data mechanism can be made more formal. Following Barber [1], split a variable $\mathbf{x}$ into observed and missing components, $\mathbf{x}^o$ and $\mathbf{x}^m$ respectively, and let $\mathbf{m}$ be a binary vector, where each element equals one when the state of the corresponding element in $\mathbf{x}$ is unknown. When data is missing, we specify a joint model over $\mathbf{x}^o$ and $\mathbf{m}$,

$$p(\mathbf{x}^o, \mathbf{m}; \boldsymbol{\theta}) = \int p(\mathbf{x}^o, \mathbf{x}^m, \mathbf{m}; \boldsymbol{\theta}) \, \mathrm{d}\mathbf{x}^m \tag{5.1}$$

$$= \int p(\mathbf{m} \mid \mathbf{x}^o, \mathbf{x}^m; \boldsymbol{\theta}) p(\mathbf{x}^o, \mathbf{x}^m; \boldsymbol{\theta}) \, \mathrm{d}\mathbf{x}^m. \tag{5.2}$$

The term $p(\mathbf{m} \mid \mathbf{x}^o, \mathbf{x}^m; \boldsymbol{\theta})$ is our model of how the missingness of a datapoint depends on the observed and missing values, as well as the parameters $\boldsymbol{\theta}$. It is through this model that we express our beliefs about how missing data is generated.

We might assume that $\mathbf{m}$ does not in fact depend on the data or parameters

$$p(\mathbf{m} \mid \mathbf{x}^o, \mathbf{x}^m; \boldsymbol{\theta}) = p(\mathbf{m}). \tag{5.3}$$

This is known as Missing Completely at Random (MCAR)[47]. A weaker assumption is that M depends only on the observed values

$$p(\mathbf{m} \mid \mathbf{x}^o, \mathbf{x}^m; \boldsymbol{\theta}) = p(\mathbf{m} \mid \mathbf{x}^o), \tag{5.4}$$

which is known as the Missing at Random (MAR) assumption. Both of these assumptions allow us to factorise Equation 5.2 as follows

$$p(\mathbf{x}^o, \mathbf{m}; \boldsymbol{\theta}) = p(\mathbf{m} \mid \mathbf{x}^o) \int p(\mathbf{x}^o, \mathbf{x}^m; \boldsymbol{\theta}) \, \mathrm{d}\mathbf{x}^m. \tag{5.5}$$

This factorisation is important for MLE, since it allows us to ignore the missing data model $p(\mathbf{m} \mid \mathbf{x}^o)$ when learning $\boldsymbol{\theta}$. The same is true in NCE / VNCE. Replacing the normalised model with a unnormalised one, ==>

$$\phi(\mathbf{x}^o, \mathbf{m}; \boldsymbol{\theta}) \propto \phi(\mathbf{m} \mid \mathbf{x}^o) \int \phi(\mathbf{x}^o, \mathbf{x}^m; \boldsymbol{\theta}) \, \mathrm{d}\mathbf{x}^m. \tag{5.6}$$

$$\propto \int \phi(\mathbf{x}^o, \mathbf{x}^m; \boldsymbol{\theta}) \, \mathrm{d}\mathbf{x}^m, \tag{5.7}$$

and hence learning is unaffected by the mechanism by which data is missing.

If we did not assume either MCAR or MAR, then the missing data mechanism could not be ignored when estimating the parameter $\boldsymbol{\theta}$. We do not consider this setting in the remaining chapter, and so refer the reader to Rubin [47] for more details.

5.1.2 The VNCE objective

In the work to follow, we assume that data is Missing Completely at Random. Specifically, we assume that, given n data points with d features, then $p \times n \times d$ randomly selected values are missing, where $0 < p < 1$. Therefore, each data point, $\mathbf{x}_i = (\mathbf{x}_i^o, \mathbf{x}_i^m)$, will have its own, potentially empty, set of missing values.

VNCE requires that the noise samples have the same dimension as the observed data, so we generate ν noise samples $\mathbf{y}_i^o$ with the same pattern of missingness as $\mathbf{x}_i$, so that $|\mathbf{y}_i^o| = |\mathbf{x}_i^o|$. The VNCE objective function, *for this datapoint*, is given by

$$J^i_{\text{VNCE}}(\boldsymbol{\theta}, q) = \mathbb{E}_{\mathbf{x}_i^m \sim q(\mathbf{x}_i^m \mid \mathbf{x}_i^o)} \log \left(\frac{\phi(\mathbf{x}_i^o, \mathbf{x}_i^m; \boldsymbol{\theta})}{\phi(\mathbf{x}_i^o, \mathbf{x}_i^m; \boldsymbol{\theta}) + \nu q(\mathbf{x}_i^m \mid \mathbf{x}_i^o) p_{\mathbf{y}_i}(\mathbf{x}_i^o)} \right) \tag{5.8}$$

$$+\nu \mathbb{E}_{\mathbf{y}_i^o} \log \left(\frac{\nu p_{\mathbf{y}_i}(\mathbf{y}_i)}{\nu p_{\mathbf{y}_i}(\mathbf{y}_i) + \mathbb{E}_{\mathbf{y}_i^m \sim q(\mathbf{y}_i^m \mid \mathbf{y}_i^o)} \left[\frac{\phi(\mathbf{y}_i^o, \mathbf{y}_i^m; \boldsymbol{\theta})}{q(\mathbf{y}_i^m \mid \mathbf{y}_i^o)} \right]} \right).$$

The full VNCE objective is then given by the average over all per-datapoint objectives

$$J_{\text{VNCE}}(\boldsymbol{\theta}, q) = \frac{1}{n} \sum_{i=1}^{n} J^i_{\text{VNCE}}(\boldsymbol{\theta}, q). \tag{5.9}$$

One important subtlety in the definition of this objective is that there is not just one noise distribution $p_{\mathbf{y}}$ or variational distribution q. Instead we require a different noise and variational distribution for each of the $2^d - 1$ non-trivial patterns of missingness that can occur in the data. This combinatorial explosion in the number of required q distributions is a problem for standard variational inference too, and is a matter of ongoing research.

One simple solution is to specify a joint noise distribution and joint variational distribution over *all* variables, missing and observed. So long as we can easily calculate,

in closed-form, all marginals of the joint noise and all conditionals of the joint variational distribution, then we can evaluate the VNCE objective.

Requiring closed-form expressions for all conditionals of a joint variational distribution can be quite restrictive. It is common in the variational inference literature to use inference networks (Section 2.5.1), which are single deterministic mappings from the observed data to parameters of the variational distributions. It is not obvious how to use a single network of this sort to model all the possible conditional distributions [55]. It is therefore an important open question as to how we might leverage inference networks to learn from missing data.

5.1.3 Cumulative data imputation

We here suggest an approximative method for filling in the data during learning that only requires d conditional distributions i.e. one per dimension. Each of these d conditionals could be implemented using an inference network. The method, which we refer to as Cumulative Data Imputation (CDI), is stated in the context of VNCE in Algorithm 1. We emphasise however, that CDI also applies when using standard variational inference for normalised models.

A simplified version of the algorithm is as follows. Suppose that our data set contains a single point with multiple missing values. CDI first fills in all missing entries with an initial guess. We then proceed by asynchronously updating the missing values—one at a time—throughout learning. Each 'epoch' consists of randomly selecting one of the missing entries of our datapoint to be treated as unobserved, whilst the others are treated as observed. Thus, the variational distribution $q(x_i^m \mid \mathbf{x}_i^o)$ is always one-dimensional and so, in total, we need at most d variational distributions. This means that the VNCE objective for missing data (5.8) is now tractable. When optimising the VNCE objective, we typically draw samples from q. These samples can now be used to update our estimate of the missing value x_i^m, by averaging them.[1] By cumulatively updating the missing entries of our dataset in this way, we conjecture that our one-dimensional conditionals can eventually learn to impute the missing values with reasonable accuracy. This iterative process of updating the missing values must also be applied to our noise samples.

Algorithm 1 requires a pre-specified function VNCEUPDATE, which updates the model parameters, $\boldsymbol{\theta}$, and a univariate conditional distribution, q_j, using the objective in

[1] We could also consider randomly selecting a sample.

Algorithm 1 Cumulative data imputation (CDI) algorithm for VNCE

Require: Model $\phi(\mathbf{u};\boldsymbol{\theta})$, and d conditional distributions $q_j(u_j \mid \mathbf{u}_{\setminus j})$.
Require: A function VnceUpdate($\mathbf{x}_{\setminus j}$, $\mathbf{y}_{\setminus j}$, $\mathcal{X}_j$, $\mathcal{Y}_j$), which updates $\boldsymbol{\theta}$ and q_j.
Require: Data set X, with observed and missing subsets X^o & X^m.
Require: Noise samples Y, with observed and missing subsets Y^o & Y^m.
Require: $\mathbf{x}_{\text{mean}} \in \mathbb{R}^d$, the means over each dimension of X^o.
Require: $\mathbf{y}_{\text{mean}} \in \mathbb{R}^d$, the means over each dimension of Y^o.
  Initialise each missing value in X^m to the value from $\mathbf{x}_{\text{mean}}$ with the same dimension.
  Repeat for Y^m, initialising with $\mathbf{y}_{\text{mean}}$.
  for i in NumEpochs **do**
    for $\mathbf{x} = (\mathbf{x}^o, \mathbf{x}^m)$ in X **do**
      Get corresponding noise vector $\mathbf{y}$.
      Select a variable x_j from the missing vector $\mathbf{x}^m$.
      Sample i.i.d. $\mathcal{X}_j = \{x_j^1, \ldots, x_j^k\} \sim q_j(x_j \mid \mathbf{x}_{\setminus j})$.
      Sample i.i.d. $\mathcal{Y}_j = \{y_j^1, \ldots, y_j^k\} \sim q_j(y_j \mid \mathbf{y}_{\setminus j})$.
      Apply VnceUpdate($\mathbf{x}_{\setminus j}$, $\mathbf{y}_{\setminus j}$, $\mathcal{X}_j$, $\mathcal{Y}_j$).
      $x_j \leftarrow \frac{1}{k}\sum_{i=1}^k x_j^k$.
      $y_j \leftarrow \frac{1}{k}\sum_{i=1}^k y_j^k$.
    end for
  end for

Equation 5.8, approximating all expectations with Monte Carlo estimates. The update to $\boldsymbol{\theta}$ will typically use gradient ascent. The update to q_j could either be 'exact', if we know the model's posterior $p(u_j|\mathbf{u}_{\setminus j};\boldsymbol{\theta})$, or use gradient ascent on some variational parameters $\boldsymbol{\alpha}_j$.

For ease of exposition, we assume VnceUpdate uses a single data point $\mathbf{x}$ and noise sample $\mathbf{y}$ to perform the updates. Generalising to larger batches is straightforward. VnceUpdate takes four arguments. A datapoint $\mathbf{x}_{\setminus j}$, which is $\mathbf{x}$ with one dimension $x_j \in \mathbf{x}^m$ missing; the corresponding noise sample $\mathbf{y}_{\setminus j}$; and samples $\mathcal{X}_j = \{x_j^1, \ldots x_j^k\}$, $\mathcal{Y}_j = \{y_j^1, \ldots y_j^k\}$ of the missing values from a variational distribution $q_j(u_j \mid \mathbf{u}_{\setminus j})$.

We show empirically that CDI is an effective method for parameter estimation in Section 5.3. We demonstrate that it works in two important cases. The first case is when the model's univariate conditionals are available, allowing us to use an EM optimisation scheme. The second case is when we approximate the conditionals with a variational family, and use a variational EM optimisation scheme.

Although CDI is really an approximation designed to cope with a combinatorial explosion in the number of required conditionals, it may have certain advantages. It can

leverage inference networks, which cannot be easily applied to missing data problems. The conditionals are one-dimensional, so that sampling is generally fast. Moreover, univariate distributions are more amenable to the reparameterisation trick. This is because we can often efficiently apply inverse transform sampling, which relies on a deterministic mapping of samples from a uniform distribution. Even sampling schemes such as rejection sampling, which are useful in one-dimension, are possible to combine with the reparameterisation trick [37]. Finally, there are models like the truncated Gaussian for which we cannot obtain exact expressions for arbitrary conditional densities, but *can* obtain closed-form expressions for the univariate conditionals. In this case, we no longer need to use approximate posteriors with variational parameters, but instead adopt an EM-type of approach. We discuss this idea in more detail in Section 5.2.3.

We see two obvious issues with CDI. The first is that we no longer have an objective function that can be evaluated on a validation or test set containing more than one missing value per datapoint. This makes hyperparameter tuning and model evaluation difficult. Secondly, we do not (currently) have any theoretical guarantees that the algorithm will converge or yield good estimates of the model's parameters. It would be interesting to analyse whether Gibbs sampling using the learned d conditionals corresponds to sampling from a meaningful joint distribution over all variables. Such an analysis could be used to understand the algorithms convergence properties and the Gibbs sampling itself might enable us to impute missing values in test data.

5.1.4 Work related to CDI

There is a relatively small literature on applying inference networks to incomplete data. All such work differs from our setting in that they use standard variational inference in the context of a variational auto-encoder (VAE). Vedantam et al. [54] address the issue of how to use inference networks to parametrise $q(\mathbf{z}|\mathbf{x}^o)$, where $\mathbf{z}$ is *not* the missing variables, but separate latent variables used in a VAE. They do not treat the missing values $\mathbf{x}^m$ as latent variables. Motivated by the work of Williams and Nasrabadi [55], they factorise q as

$$q(\mathbf{z} \mid \mathbf{x}^o) = \prod_{i=1}^{|\mathbf{x}^o|} q_i(\mathbf{z} \mid x_i^o), \tag{5.10}$$

where each of the d variational distributions q_i is a multivariate Gaussian with mean and covariance parametrised by an inference network. The resulting product is also Gaussian, and can be stated in closed-form. It is tempting to adapt their work to handle missing data by using

$$q(\mathbf{x}^m \mid \mathbf{x}^o) = \prod_{i=1}^{|\mathbf{x}^o|} q_i(\mathbf{x}^m \mid x_i^o), \tag{5.11}$$

however using d inference networks with fixed output dimension will not work, since the dimension of $\mathbf{x}^m$ varies across data points.

Concurrently with this work, Nazabal et al.[38] proposed a method for using inference networks that, like CDI, treats the missing values as latent variables. Their work was also in the context of a VAE, and used the following factorisation

$$q(\mathbf{z}, \mathbf{x}^m \mid \mathbf{x}^o) = q_0(\mathbf{z} \mid \mathbf{x}^o) \prod_{i=1}^{|\mathbf{x}^m|} q_i(x_i^m \mid \mathbf{z}), \tag{5.12}$$

which assumes that each variable x_i^m is conditionally independent of all other missing and observed variables given $\mathbf{z}$. This differs from our method, which is designed to work for models without additional latent variables $\mathbf{z}$.

5.2 Learning graphical models with incomplete data

In this section, we demonstrate how VNCE can be used to train a truncated Gaussian model with a sparse precision matrix from incomplete data. By treating the missing values as latent variables and performing approximate inference over them, we offer a probabilistically principled strategy for 'filling-in' the unobserved data. By learning a sparse precision matrix, we can then derive an undirected graph describing the conditional independence relationships between variables.

Our work here builds on that of Lin et al.[30], who used an L1-regularised version of the (non-negative) score matching objective [22] to train Gaussian models, truncated at zero along all dimensions. Truncations like this are useful for injecting prior knowledge into a model; many scientific domains are governed by physical laws that impose strict constraints on what values the data can take. The most obvious and widespread instance of this being non-negativity.

Lin et al. [30] fit this truncated Gaussian to RNA-Seq data, which quantifies the amount of RNA molecules present in a population of cells. This set of RNA molecules is known as the *transcriptome*, and it reflects which genes are actively expressed (and hence which are suppressed) at a given moment in time. RNA-Seq data can be used to analyse the hundreds of gene interactions driving complex diseases such as cancer, making it an important application area of large-scale graphical modelling.

As is the case for many real-world data sets, a significant fraction of the RNA-Seq data was missing. Lin et al[30] use a data set with 350 features (which correspond to genes), but report that 5% of these features have more than 10% of their values missing. They therefore discard these features. For all remaining features, any missing values were set to zero. They comment on this issue in the discussion, suggesting that it ought to be possible to develop a version of score matching to handle missing data. Whilst we agree that this is a promising direction for future research, such a technique remains to be developed. Thus, we here use VNCE instead.

Setting missing values to zero is not optimal. Ideally, we would use the observed values to infer the missing ones, under the assumption that correlations exist between them. One strategy is to impute missing features with the average value of that feature across the observed data. A more sophisticated approach is to use a regression model to map observed features to unobserved ones[31]. Perhaps the most principled approach, however, it is to build a probabilistic model over all features and use conditional distributions to infer missing values. It is this latter approach that we use when applying VNCE.

We now review the necessary background material for learning Gaussian undirected graphical models, and discuss how to extend this to truncated Gaussians. It is important to emphasise that a truncated Gaussian is just one example application for VNCE—It is typically the case that undirected graphical models have intractable partition functions and may therefore benefit from VNCE when data is missing.

5.2.1 Gaussian undirected graphical models

Recall that the pdf for the multivariate Gaussian is proportional to:

$$p(\mathbf{x}; \Sigma, \mu) \propto \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \tag{5.13}$$

where $\boldsymbol{\mu} \in \mathbb{R}^d$ is the mean and $\Sigma \in \mathbb{R}^{d \times d}$ is the positive semi-definite covariance matrix. To see how a multivariate Gaussian can be viewed as an undirected graphical model, consider the following factorisation of its probability density function:

$$p(\mathbf{x}; \Sigma, \mu) \propto \prod_{i=1}^{d} \exp\left(-\frac{1}{2} K_{i,i} x_i^2 + b_i x_i\right) \prod_{0 \leq i < j \leq d} \exp\left(-K_{i,j} x_i x_j\right), \tag{5.14}$$

where $K = \Sigma^{-1}$ and b_i is the i^{th} element of $\mathbf{b} = K\boldsymbol{\mu}$. The first d factors of the left-most product each involve only a single x_i, and so do not imply any dependencies between the variables. The remaining factors contain products of all *pairs* of variables, x_i and x_j. If $K_{i,j}$ is non-zero, there will be an edge between x_i and x_j in the undirected graph associated to the model.

In general, when we estimate K from data, all entries will be non-zero and so the associated graph will be fully-connected. This can even occur when two variables are genuinely conditionally independent given the other variables in the graph. If our primary objective is to learn about these conditional independencies, then it helps to 'push' the off-diagonal elements of K towards zero. We can do this by adding an L1-regularisation term to our objective function, $\sum_{i<j} |K_{i,j}|$, where, since K is symmetric, it is sufficient to sum over the lower triangular off-diagonal elements.

5.2.2 Truncated Gaussians

Gaussians have positive density over their entire domain. In contrast, truncated Gaussians have zero density over regions of the input space. Probability mass has been 'chopped off', which means that the remaining density integrates to less than one, rendering the model unnormalised.

The unnormalised pdf is given by:

$$\phi(\mathbf{x}; K, \boldsymbol{\mu}) = \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T K (\mathbf{x} - \boldsymbol{\mu})\right) \mathbb{I}(\mathbf{x} \in A), \tag{5.15}$$

where $A \in \mathbb{R}^d$ is some subset of the input space.[2] We assume throughout the rest of the chapter that the set A is the positive orthant: $[0, +\infty]^d$. Because the pdf has this form, the preceding section on Gaussian graphical models applies equally to truncated

[2]Technically, the set A needs to satisfy some measure-theoretic conditions, but all reasonable choices will meet these conditions.

Gaussians, and so learning the matrix K automatically gives us an undirected graph from which we can read off conditional independence relationships.

For almost all choices of parameters and sets A, we cannot normalise $\phi(\mathbf{x}; K, \boldsymbol{\mu})$ since the integral $\int_A \phi(\mathbf{x}; K, \boldsymbol{\mu})\, d\mathbf{x}$ is intractable, except in very low dimensions [20]. It is therefore necessary to learn the parameters using a method for unnormalised models such as NCE, rather than maximum likelihood estimation. When applying NCE to estimate the parameters K and $\boldsymbol{\mu}$, we multiply the model by a scaling parameter, $\exp(c)$, and so a cleaner expression for ϕ is

$$\phi(\mathbf{x}; \boldsymbol{\theta}) = \exp\left(-\frac{1}{2}\mathbf{x}^T K \mathbf{x} + \mathbf{x}^T \mathbf{b} + c\right) \mathbb{I}(\mathbf{x} \in [0, +\infty]^d), \tag{5.16}$$

where $\boldsymbol{\theta} \equiv (K, \mathbf{b}, c)$, $\mathbf{b} = K\boldsymbol{\mu}$ and we have absorbed $-1/2\boldsymbol{\mu}^T K \boldsymbol{\mu}$ into the scaling parameter.

When our data set contains missing values, evaluating the model ϕ at an observed datapoint $\mathbf{x}$ is no longer possible in general. We would have to integrate over the missing values, which is intractable when $\mathbf{z}$ is more than a few dimensions. To handle this intractability, we could apply VNCE instead of NCE. VNCE, in its basic form, requires a conditional distribution for every subset of missing variables. We might consider using the model's conditionals, but these are also truncated normals [20], which we cannot normalise in high-dimensions. Since VNCE requires normalised conditionals, this strategy will not work. Instead, we propose to use CDE (Section 5.1.3), which only requires *univariate* conditionals. These can be derived from Equation 5.16 in closed form. Suppose we have observed all but the j^{th} variable, and $\mathbf{x}_{\setminus j} = \boldsymbol{a}$. Then,

$$p(x_j \mid \mathbf{x}_{\setminus j} = \boldsymbol{a}; \boldsymbol{\theta}) = \frac{\psi(\frac{x_j - \bar{\mu}_j}{\bar{\sigma}_j})}{\left(1 - \Phi(\frac{-\bar{\mu}_j}{\bar{\sigma}_j})\right)\bar{\sigma}_j}, \tag{5.17}$$

where ψ is the pdf of a standard normal, Φ its cumulative density function (cdf), and $\bar{\mu}_j$ and $\bar{\sigma}_j$ are given by

$$\bar{\sigma}_j^2 = \frac{1}{K_{j,j}}, \qquad \bar{\mu}_j = \mu_j + \bar{\sigma}_j^2 K_{j,\setminus j}(\boldsymbol{\mu}_{\setminus j} - \boldsymbol{a}). \tag{5.18}$$

Both K and $\boldsymbol{\mu}$ were defined in Equation 5.15, and the notation $K_{j,\setminus j}$ denotes the j^{th} row of K with the j^{th} element removed. A proof of this result is provided in Appendix A.4.

5.2.3 Applying VNCE to a truncated Gaussian

We here outline our choices for the components required by VNCE in the context of the truncated Gaussian model.

Model. We do not enforce the constraint that the matrix K in Equation 5.16—is semi-positive definite, only that it is symmetric. This is in line with previous work [30, 25]. The idea is that, since we are only interested in graph learning, it makes sense to optimise within a less constrained space. It is worth noting that, technically, the resultant model is not really Gaussian and may not always be normalisable i.e. the partition function may not be finite for some parameter values[3].

Variational distribution. We consider two choices of univariate conditional distributions that can then be plugged into the CDI algorithm. The first choice is the model's conditionals given by Equations 5.17 and 5.18. The second choice is a variational family with the same form as Equation 5.17, but the parameters μ_j and σ_j are given by affine transformations of the observed values $\mathbf{x}_{\setminus j} = \mathbf{a}$,

$$\mu_j = \boldsymbol{v}^T\mathbf{a} + c_j, \qquad \sigma_j = \boldsymbol{w}^T\mathbf{a} + d_j, \tag{5.19}$$

where $\boldsymbol{v}_j, \boldsymbol{w}_j, c_j, d_j$ are parameters that we learn by optimising the VNCE objective.

This variational family contains the model's posterior, as we see by comparing Equations 5.19 and 5.18. VNCE with the model's conditionals should outperform VNCE with the parametrised family, since the latter tries to approximate the former. We consider both in order to determine what performance loss, if any, is incurred by having to learn an approximate posterior.

Both of our choices for the variational distribution are simple to sample from using inverse transform sampling [9],

$$x_j = \Phi^{-1}\left(\Phi\left(\frac{-\mu_j}{\sigma_j}\right) + U\left(1 - \Phi\left(\frac{-\mu_j}{\sigma_j}\right)\right)\right)\sigma_j + \mu_j, \qquad U \sim \mathcal{U}(0,1). \tag{5.20}$$

Moreover, it is straightforward to apply the reparameterisation trick when backpropagating gradients to optimise the variational parameters in Equation 5.19, since x_j is differentiable with respect to μ_j and σ_j.

[3]This is not typically an issue in the NCE framework, since we do not estimate the partition function during learning. The scaling parameter in NCE only approximates the value of the partition function for the final parameter obtained at the end of learning.

Noise. We use a noise distribution that fully factorises into a product of univariate truncated Gaussians whose means and variances are estimated from the observed data. This noise has the nice property that we can easily evaluate the marginal of any subset of the variables, which is necessary since different subsets of variables are observed per-datapoint. Estimating the means and variances requires care due to numerical stability issues not handled by many statistical software libraries; we therefore give full details of our procedure.

Given a design matrix X containing our data, we want to fit a univariate truncated Gaussian to each column. To do so, we could estimate the means μ_i and variances σ_i^2 of the *pre-truncated* Gaussians using the following equations [9], where x_i denotes a column of X with empirical mean $\bar{\mu}_i$ and variance $\bar{\sigma}_i^2$:

$$\bar{\mu}_i = \mu_i + \frac{\psi(\alpha)}{1-\Phi(\alpha)}\sigma_i, \qquad \bar{\sigma}_i^2 = \left[1 + \frac{\alpha\psi(\alpha)}{1-\Phi(\alpha)} - \left(\frac{\psi(\alpha)}{1-\Phi(\alpha)}\right)^2\right]\sigma_i^2. \tag{5.21}$$

As before, ψ is the pdf of a standard normal and Φ is its cdf. These pairs on non-linear simultaneous equations can then be solved with a variety of methods, such as Newton-Krylov [28]. However, whenever $\alpha = \frac{-\mu_i}{\sigma_i} \gg 0$, computing the fractions $\frac{\alpha\psi(\alpha)}{1-\Phi(\alpha)}$, $\frac{\psi(\alpha)}{1-\Phi(\alpha)}$ becomes numerically unstable. In a short note available on GitHub, Fernandez-de-cossio Diaz [13] explains how to fix this using the more numerically stable scaled complementary error function $\mathrm{erfcx}(x) = \exp(x^2)\,\mathrm{erf}(x)$, where $\mathrm{erf}(x)$ is the error function. Introducing the notation

$$F_1(x) = \frac{1}{\mathrm{erfcx}(x)}, \qquad F_2(x) = \frac{x}{\mathrm{erfcx}(x)}, \tag{5.22}$$

we can then re-express the required fractions in a numerically stable form,

$$\frac{\alpha\psi(\alpha)}{1-\Phi(\alpha)} = \frac{2}{\sqrt{\pi}}F_2(\frac{\alpha}{\sqrt{2}}), \qquad \frac{\psi(\alpha)}{1-\Phi(\alpha)} = \frac{2}{\sqrt{\pi}}F_2(\frac{\alpha}{\sqrt{2}}) - \frac{2}{\pi}\left[F_1(\frac{\alpha}{\sqrt{2}})\right]^2. \tag{5.23}$$

5.3 Simulations

We test the ability of VNCE to recover the ground-truth parameters $K \in \mathbb{R}^{d\times d}$ and $\mathbf{b} = K\boldsymbol{\mu} \in \mathbb{R}^d$ of a truncated normal as given in Equation 5.16 for different amounts of missing data. We are interested in sparse matrices K that correspond to graphs with many missing links. Our primary objective is to learn this graph structure and thereby infer conditional independence relationships between the d variables. We are

also interested in evaluating the accuracy of the learned parameters, which indicates the quality of our estimated probability density function.

5.3.1 Experimental setup

We generate 50 sets of parameters for $d = 5$ as follows. All elements of the main diagonal of K are set to one, whilst all elements of the superdiagonal[4] are sampled from $\mathcal{U}(0.3, 0.5)$. The top right-hand corner is also sampled from the same uniform distribution. This matrix, $\hat{K}$, is then symmetrised by setting $K = \hat{K} + \hat{K}^T - \text{diag}(\hat{K})$. The resultant graph associated to K has a ring-structure as shown in Figure 5.1. Finally, all elements of the vector $\boldsymbol{\mu}$ are set to one.

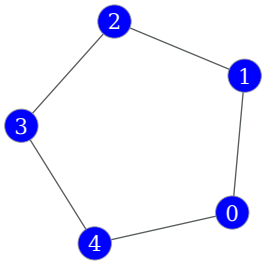

Fig. 5.1 The graph associated to the truncated normal graphical model from which we simulate data. Each node corresponds to a random variable, and the pattern of edges implies that each variable is independent of the rest given its two neighbours.

For each of the 50 sets of parameters, we sample 1000 data points. Since we are in low dimensions, sampling from the multivariate truncated normal can be achieved with rejection sampling with a Gaussian proposal distribution. For higher dimensions, more sophisticated techniques exist [8]. For each data set, we generate 10 boolean masks, such that a '1' corresponds to a missing data point. The fraction of ones in each mask increases in increments of 10%, ranging from 0% to 90%. Applying these masks to the data, we obtain a total of 500 data sets.

Methods for comparison

We apply the CDI algorithm for VNCE (Algorithm 1), for both choices of variational distribution described in Section 5.2.3. We compare to baseline methods that fill in the missing values once at the beginning of learning and then use NCE (Section 2.6),

[4] The 'superdiagonal' refers to those elements directly above the main diagonal.

treating the data as fully observed. This is line with previous work, where Lin et al. [30] fill in missing values with zeros before estimating a truncated normal. The primary difference is that they apply score matching rather than NCE.

In total, we compare five methods, all using the same noise distribution given in Section 5.2.3. All methods use stochastic gradient ascent (SGA) with a mini-batch of size 100 and a learning rate of 0.1. The methods are:

- **VNCE (true)**. Uses the true posteriors of the model, learning with the EM algorithm for VNCE. In the M-step of learning we update the model's parameters using SGA.
- **VNCE (approx)**. Uses an approximate posterior, learning with the variational EM algorithm, alternating between two optimisation steps: one for the model parameters $\boldsymbol{\theta}$ and one for the variational parameters $\boldsymbol{\alpha}$. Each step uses SGA.
- **NCE (means)**. Fills in with the means of the observed data features.
- **NCE (noise)**. Fills in with samples from the noise distribution.
- **NCE (random)**. Fills in with samples drawn from $\mathcal{U}(0, 3)$.

Finally, we note that whilst Lin et al.[30] used L1-regularisation to help learn a sparse precision matrix, we here consider the simpler case without an L1 penalty. This is to make analysing the performance of VNCE and CD simpler. In future work we intend to see what additional impact regularisation has.

Performance metrics

We consider two performance metrics: mean squared error (MSE) between the estimated and true parameters, and ROC curves for measuring how well we learn the graph structure in Figrue 5.1.

For the mean squared error, we split the parameters of the truncated normal (Equation 5.16) into three components: $\mathbf{b} = K\boldsymbol{\mu}$, the diagonal of K, and the lower-triangular off-diagonal elements of K. We use $\mathbf{b}$, rather than $\boldsymbol{\mu}$, because $\boldsymbol{\mu}$ only affects the shape of our model indirectly, through $\mathbf{b}$. We split K into two components since the off-diagonal elements capture the interactions between variables, which is important for graphical modelling, whilst the diagonal terms relate to each variable individually.

After parameter estimation, we infer a graph from the off-diagonal elements of K, and then assess what fraction of the true edges we obtain (true positives) and what

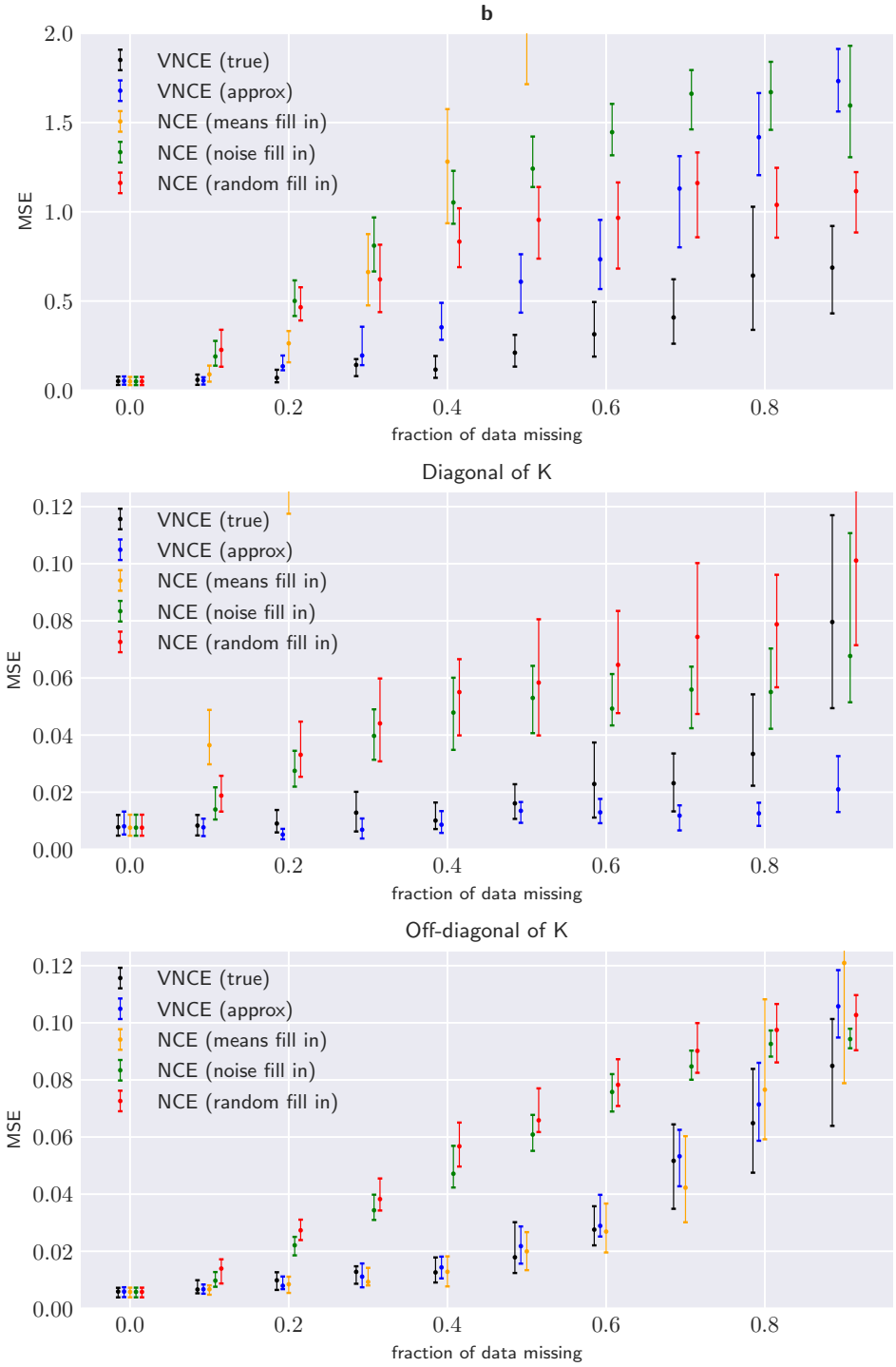

Fig. 5.2 Mean squared error of estimated parameters of a truncated Gaussian for different fractions of missing data. The vertical bars represent interquartile ranges, whilst the circles are median values. In both the first and second plot, the orange bar is omitted in many cases, since the MSEs were too large to include. Overall, we see that either VNCE (true) or VNCE (approx) has the lowest MSE, or joint lowest, across almost all fractions of missing data. This shows that probabilistic inference of missing values improves parameter estimation

fraction of the false edges we obtain (false positives). One issue is that our estimated matrix K is generally non-zero for all elements, implying a fully-connected graph. To resolve this, we apply a threshold, such that if the absolute value of an element is below that threshold, then the corresponding edge is not included in the graph. For different choices of threshold, we obtain a different trade-off between true positives and false positives. This trade-off is depicted by the Receiver Operating Characteristic curve (ROC curve).

5.3.2 Results

Accuracy of parameter estimation

Figure 5.2 shows the simulation results for different fractions of missing data, $0 < p < 1$, plotted against the mean squared error (MSE) of the parameters. The vertical bars display the interquartile ranges over the 50 simulations for each method.

Either VNCE(true) or VNCE(approx) has the lowest MSE, or joint lowest, across almost all parameters and fractions of missing data. This shows that probabilistic inference of missing values improves parameter estimation. Moreover, the performance gains from VNCE typically grow with the fraction of missing data until around 50% is missing.

For the $\mathbf{b}$ parameter, we see that VNCE(true) consistently has the lowest MSE for more than 10% missing. VNCE(approx) is the second best method up until 70% is missing, at which point it is no better than baseline methods. For the diagonal of K, the reverse trend holds. VNCE(approx) has the lowest MSE, whilst VNCE(approx) is second best, up until 90% of data is missing. The reason for these opposing trends is not entirely clear. We might expect VNCE(true) to be consistently better, since it does not rely on a variational approximation. However, it may be the case that the approximation simply makes different trade-offs, fitting some parameters more accurately than others. Moreover, it is unclear how the CD learning algorithm interacts with the two types of posteriors, making interpretation difficult.

Both VNCE methods and NCE(means) have comparable performance for the off-diagonal elements of K, although VNCE(true) generally has a slightly lower median MSE, particularly when more than 80% of the data is missing. It is interesting that NCE(means) performs comparably. It seems that filling in each missing dimension with the same fixed mean allows NCE to still learn the interactions between variables. This does come at a cost however. NCE(means) provides poor estimates of the diagonal of

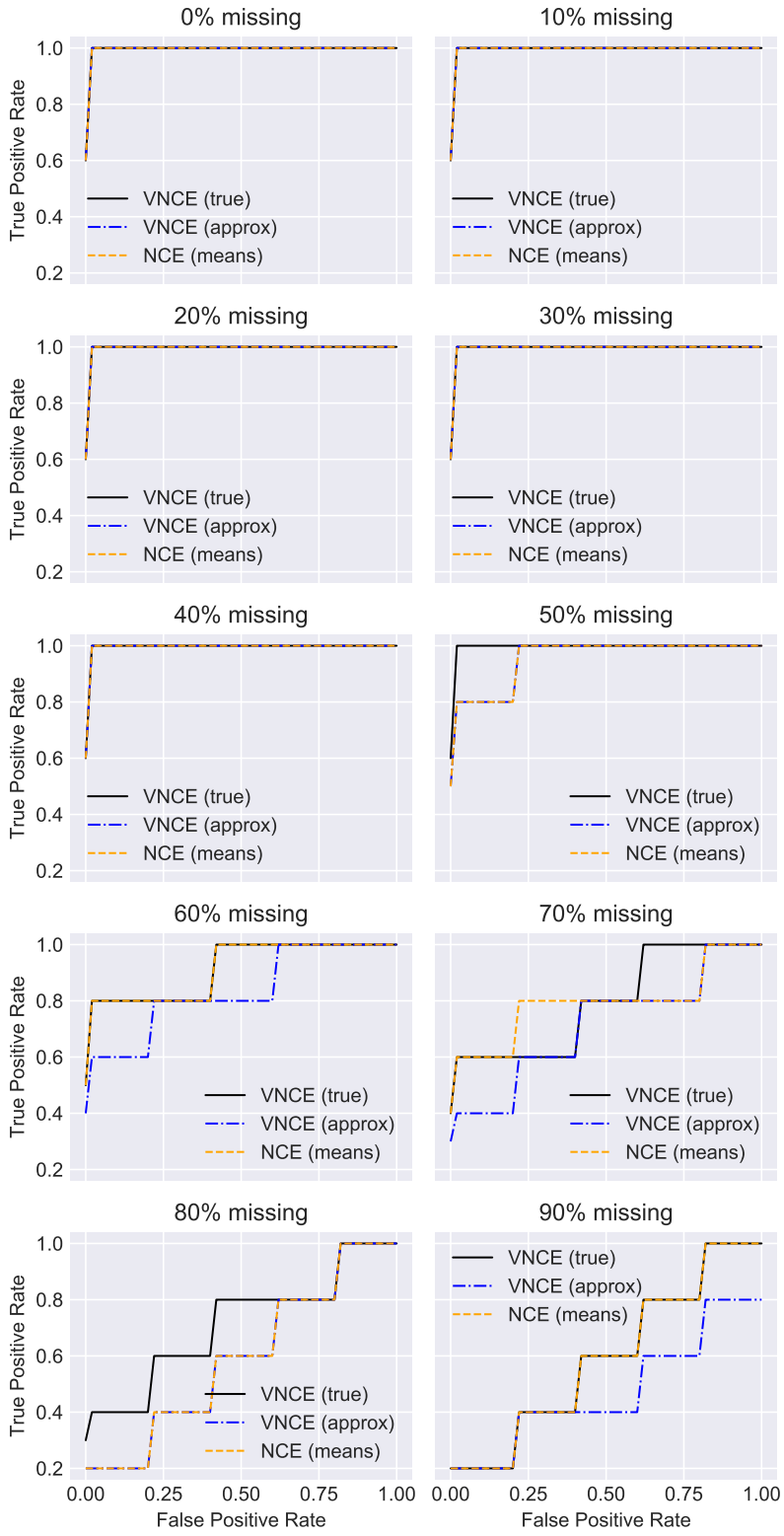

Fig. 5.3 Median ROC curves measuring the ability of the three best methods to recover the ground-truth conditional independence relationships between variables. Moving row-by-row, from left-to-right, the fraction of missing data increases in increments of 10%. For less than 50% missing, all methods perfectly recover the true independence relationships. For higher amounts of missing data, VNCE (true) is best, followed by NCE (means) and then VNCE (approx).

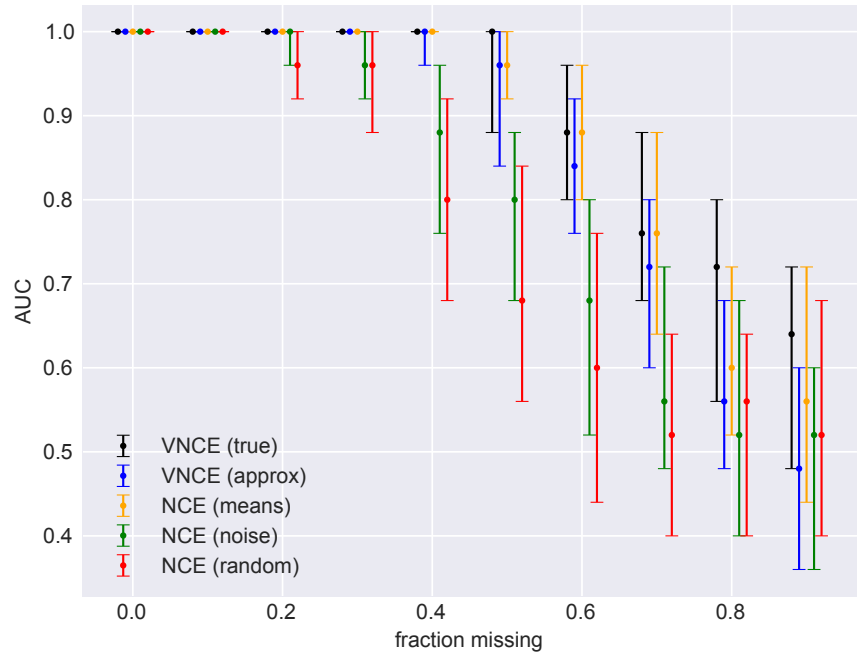

Fig. 5.4 Interquartile ranges of the AUC scores for each of the five methods when trying to recover the ground-truth conditional independence relationships between variables. Larger AUC values correspond to better performance. VNCE(true) and NCE(means) are the joint best methods when less than 80% of data is missing. After this point, VNCE(true) obtains the highest median AUC scores.

K and the parameter b; for most fractions of missing data, their MSEs were too high to include in the plot.

Quality of the learned graphs

Figure 5.3 displays the median[5] ROC curves for the top three performing methods: VNCE(true), VNCE(approx) and NCE(means). The other two baselines were omitted for clarity. We see that all three methods perfectly identify the true graph when less than 50% of data is missing. For 50% and above, the methods still perform similarly, but it seems that VNCE(true) is best, followed by NCE(means) and then VNCE(approx).

The interquartile range of the Area under the ROC Curve (AUC) for each of the 5 methods is shown in Figure 5.4. It is clear from this figure that VNCE(true) outperforms NCE(means) when the fraction of missing data is 80% or greater. In

[5]Specifically, for every false positive rate in the range $[0, 0.02, \ldots, 0.098, 1]$, we calculate the median true positive rate over all 50 simulations.

contrast, VNCE (approx) deteriorates for more than 80% missing, performing worse than all baselines. Thus, whilst VNCE can be effective, the use of a variational approximation may be problematic for large amounts of missing data.

Finally, in Figure 5.5, we show 'point estimates' of the learned graph for each of the best three methods. These estimates were generated by randomly selecting one of the 50 simulations and taking the 5 largest elements (in absolute value) from the off-diagonal of the learned matrices K and plotting the corresponding edges.

5.3.3 Limitations and future work

We have shown that VNCE can perform accurate parameter estimation and graph selection for an unnormalised graphical model in the context of missing data. However, our experiments were limited to 5 dimensional simulations, and so future experiments will need to investigate the scalability of our methods to real-world data such as the RNA-Seq data used by Lin et al [30]. Since this previous work used score matching with missing data filled in with zeros, it makes sense to include this as an additional baseline method. Moreover, it is important to assess the potential benefits of using L1-regularisation.

We introduced cumulative data imputation (CDI), a novel variational learning framework for missing data, and showed that it works empirically. However, we were unable to quantify the impact CDI had on learning. To measure this, we would need to compare our results to VNCE without CDI. This could be achieved by using a variational family for which we can compute, in closed-form, all conditional distributions. For instance, a log-normal variational family could be used.

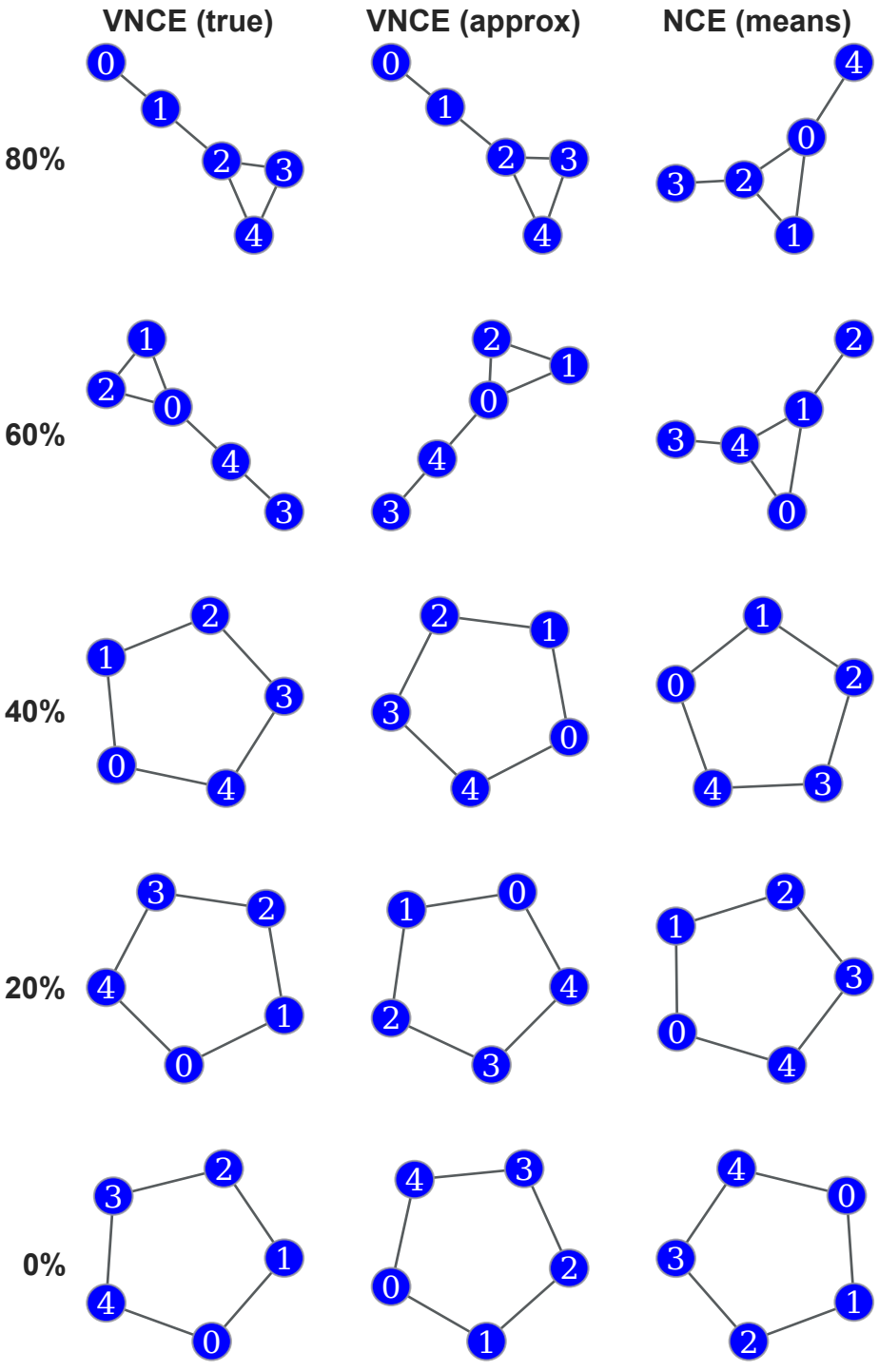

Fig. 5.5 Illustrative graphs learned when trying to recover the ground-truth conditional independence relationships between variables. These graphs were generated by taking the 5 largest elements (in absolute value) from the off-diagonal of the learned matrices K and plotting the corresponding edges.

Chapter 6

Conclusions

Our primary goal in this thesis was to develop a new method for parameter estimation of unnormalised latent variable models. This family of distributions is very broad, with the potential to approximate a diverse array of data-generating distributions. However, training unnormalised latent variable models is doubly-intractable in general, restricting the usefulness of such models in practice.

We achieved our objective by developing a method for training unnormalised latent variable models that extends noise-contrastive estimation (NCE). We derived a variational lower bound to the NCE objective, and proved that maximising this lower bound is equivalent to NCE. This theoretical achievement resolves a key deficiency in the noise-contrastive framework. Moreover, we saw that only one other method—(persistent) contrastive divergence—is able to train unnormalised, latent variable models. Given the potential power and flexibility of such models, the addition of a second viable estimation technique is an important advance.

We provided unifying theory to connect VNCE with standard variational inference on the log-likelihood. We showed that VNCE performs a type of approximate inference based on minimising an f-divergence that differs in general from the KL-divergence. This is an important contribution since learning an approximate posterior can be useful for a variety of tasks, independent of the learned model.

Finally, we demonstrated the effectiveness of VNCE empirically in the context of a truncated normal graphical model with missing data. To achieve this goal, we developed a novel approximate algorithm, cumulative data imputation (CDI), for applying variational methods to missing-data problems. This algorithm applies to standard variational inference, not just VNCE, and allows us to leverage inference networks to impute missing values during learning.

Whilst we have laid the foundations of VNCE, more work is needed to empirically evaluate the method on real-world, high-dimensional datasets. A key motivating goal for training unnormalised latent variable models is that they are flexible, with the potential to model complex, high-dimensional data such as natural images. It is therefore an important next step to implement VNCE at scale.

In particular, we see promise in applying both NCE and VNCE to models parametrised by deep neural networks, as was recently shown to be possible for a version of score matching [50]. The key obstacle here is choosing a good noise distribution. Recent work [11] has shown that it is possible to semi-automate the choice of noise, and we believe further advances can be made.

Finally, it would be interesting to apply CDI to standard variational inference problems for normalised models. At present, there is only one other method (to our knowledge) that treats missing data probabilistically and uses inference networks [38]. A comparison of the two methods in the context of an auto-encoder would therefore be an important next step.

References

[1] Barber, D. (2012). *Bayesian reasoning and machine learning*. Cambridge University Press.

[2] Barthelmé, S. and Chopin, N. (2015). The Poisson transform for unnormalised statistical models. *Statistics and Computing*, 25(4):767–780.

[3] Bartle, R. G. (2014). *The elements of integration and Lebesgue measure*. John Wiley & Sons.

[4] Belanger, D. and McCallum, A. (2016). Structured prediction energy networks. In *International Conference on Machine Learning*, pages 983–992.

[5] Bengio, Y. and Delalleau, O. (2009). Justifying and generalizing contrastive divergence. *Neural computation*, 21(6):1601–1621.

[6] Besag, J. (1975). Statistical analysis of non-lattice data. *The Statistician*, pages 179–195.

[7] Bonnet, G. (1964). Transformations des signaux aléatoires a travers les systemes non linéaires sans mémoire. In *Annales des Télécommunications*, volume 19, pages 203–220. Springer.

[8] Botev, Z. (2017). The normal law under linear restrictions: simulation and estimation via minimax tilting. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(1):125–148.

[9] Burkardt, J. (2014). The truncated normal distribution. *Department of Scientific Computing Website, Florida State University*.

[10] Carreira-Perpinan, M. A. and Hinton, G. E. (2005). On contrastive divergence learning. In *AISTATS*, volume 10, pages 33–40.

[11] Ceylan, C. and Gutmann, M. U. (2018). Conditional Noise-Contrastive Estimation of Unnormalised Models. *Proceedings of the 35th International Conference on Machine Learning*.

[12] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (methodological)*, pages 1–38.

[13] Fernandez-de-cossio Diaz, J. (2018). Moments of the univariate truncated normal distribution.

[14] Gershman, S. and Goodman, N. (2014). Amortized inference in probabilistic reasoning. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 36.

[15] Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

[16] Gutmann, M. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13:307–361.

[17] Gutmann, M. and Hyvärinen, A. (2013). Estimation of unnormalized statistical models without numerical integration. In *Proc Workshop on Information Theoretic Methods in Science and Engineering*.

[18] Hinton, G. E. (2006). Training products of experts by minimizing contrastive divergence. *Training*, 14(8).

[19] Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.

[20] Horrace, W. C. (2005). Some results on the multivariate truncated normal distribution.

[21] Hyvärinen, A. (2005). Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709.

[22] Hyvärinen, A. (2007). Some extensions of score matching. *Computational statistics & data analysis*, 51(5):2499–2512.

[23] Jensen, J. L. W. V. (1906). Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta mathematica*, 30(1):175–193.

[24] Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

[25] Khare, K., Oh, S.-Y., and Rajaratnam, B. (2015). A convex pseudolikelihood framework for high dimensional partial correlation estimation with convergence guarantees. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 77(4):803–825.

[26] Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. (2014). Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589.

[27] Kingma, D. P. and Welling, M. (2013). Stochastic gradient VB and the variational auto-encoder. *The 2nd International Conference on Learning Representations*.

[28] Knoll, D. A. and Keyes, D. E. (2004). Jacobian-free Newton–Krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397.

[29] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

[30] Lin, L., Drton, M., and Shojaie, A. (2016). Estimation of high-dimensional graphical models using regularized score matching. *Electronic journal of statistics*, 10(1):806.

[31] Little, R. J. and Rubin, D. B. (2014). *Statistical analysis with missing data*, volume 333. John Wiley & Sons.

[32] Lyu, S. (2009). Interpretation and generalization of score matching. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 359–366. AUAI Press.

[33] Mangel, M. and Samaniego, F. J. (1984). Abraham Wald's work on aircraft survivability. *Journal of the American Statistical Association*, 79(386):259–267.

[34] Matsuda, T. and Hyvarinen, A. (2018). Estimation of Non-Normalized Mixture Models and Clustering Using Deep Representation. *arXiv preprint arXiv:1805.07516*.

[35] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

[36] Mnih, A. and Gregor, K. (2014). Neural variational inference and learning in belief networks. *Proceedings of the 31st International Conference on Machine Learning*.

[37] Naesseth, C., Ruiz, F., Linderman, S., and Blei, D. (2017). Reparameterization gradients through acceptance-rejection sampling algorithms. In *Artificial Intelligence and Statistics*, pages 489–498.

[38] Nazabal, A., Olmos, P. M., Ghahramani, Z., and Valera, I. (2018). Handling incomplete heterogeneous data using VAEs. *arXiv preprint arXiv:1807.03653*.

[39] Neal, R. M. and Hinton, G. E. (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer.

[40] Opper, M. and Saad, D. (2001). *Advanced mean field methods: Theory and practice*. MIT press.

[41] Osindero, S. and Hinton, G. E. (2008). Modeling image patches with a directed hierarchy of Markov random fields. In *Advances in neural information processing systems*, pages 1121–1128.

[42] Paisley, J., Blei, D., and Jordan, M. (2012). Variational Bayesian inference with stochastic search. *Proceedings of the 28th international conference on Machine learning*.

[43] Price, R. (1958). A useful theorem for nonlinear devices having Gaussian inputs. *IRE Transactions on Information Theory*, 4(2):69–72.

[44] Ranganath, R., Gerrish, S., and Blei, D. (2014). Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822.

[45] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *Proceedings of the 31st International Conference on Machine Learning*.

[46] Robert, C. (2014). Machine learning, a probabilistic perspective.

[47] Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3):581–592.

[48] Ruslan, S. and Geoffrey, H. (2009). Deep Boltzmann machines. *J Mach Learn Res*, 24(5):448–455.

[49] Salakhutdinov, R. (2008). Learning and evaluating Boltzmann machines. *Tech. Rep., Technical Report UTML TR 2008-002, Department of Computer Science, University of Toronto*.

[50] Saremi, S., Mehrjou, A., Schölkopf, B., and Hyvärinen, A. (2018). Deep energy estimator networks. *arXiv preprint arXiv:1805.08306*.

[51] Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado univ at boulder dept of computer science.

[52] Sutskever, I. and Tieleman, T. (2010). On the convergence properties of contrastive divergence. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 789–795.

[53] Tieleman, T. and Hinton, G. (2009). Using fast weights to improve persistent contrastive divergence. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1033–1040. ACM.

[54] Vedantam, R., Fischer, I., Huang, J., and Murphy, K. (2017). Generative models of visually grounded imagination. *arXiv preprint arXiv:1705.10762*.

[55] Williams, C. K. and Nash, C. (2018). Autoencoders and Probabilistic Inference with Missing Data: An Exact Solution for The Factor Analysis Case. *arXiv preprint arXiv:1801.03851*.

[56] Younes, L. (1998). Stochastic gradient estimation strategies for Markov random fields. In *Bayesian inference for inverse problems*, volume 3459, pages 315–326. International Society for Optics and Photonics.

Appendix A

A.2 Gradient of $J_{\text{VNCE}}(\boldsymbol{\theta})$

$$J_{\text{VNCE}}(\boldsymbol{\theta}) = -\mathbb{E}_{\mathbf{x}}\mathbb{E}_{\mathbf{z}\sim q_k}\left[\log(\psi_1(\mathbf{x},\mathbf{z};\boldsymbol{\theta}))\right] - \nu\mathbb{E}_{\mathbf{y}}\left[\log(\psi_2(\mathbf{y};\boldsymbol{\theta}))\right], \tag{A.3}$$

where

$$\psi_1(\mathbf{x},\mathbf{z};\boldsymbol{\theta}) = 1 + \frac{\nu}{r(\mathbf{x},\mathbf{z};\boldsymbol{\theta})} \qquad \psi_2(\mathbf{y};\boldsymbol{\theta}) = 1 + \frac{1}{\nu}\mathbb{E}_{\mathbf{z}\sim q_k}[r(\mathbf{y},\mathbf{z};\boldsymbol{\theta})] \tag{A.4}$$

$$\text{and} \qquad r(\mathbf{u},\mathbf{z};\boldsymbol{\theta}) = \frac{\phi(\mathbf{x},\mathbf{z};\boldsymbol{\theta})}{q(\mathbf{z}\mid\mathbf{u})p_{\mathbf{y}}(\mathbf{u})} \tag{A.5}$$

Now, let us take derivative wrt $\boldsymbol{\theta}$:

$$\nabla_{\boldsymbol{\theta}} \log(\psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})) = \frac{1}{\psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})} \nabla_{\boldsymbol{\theta}} \psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) \tag{A.6}$$

$$= \frac{1}{\psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})} \frac{-\nu}{r(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})^2} \nabla_{\boldsymbol{\theta}} r(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) \tag{A.7}$$

$$= \frac{1}{\psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})} \frac{-\nu}{r(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})^2} \frac{\nabla_{\boldsymbol{\theta}} \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})}{q_k(\mathbf{z} \mid \mathbf{x}) p_{\mathbf{y}}(\mathbf{x})}. \tag{A.8}$$

$$\nabla_{\boldsymbol{\theta}} \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) = [\nabla_{\boldsymbol{\theta}} \log(\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}))]\, \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}). \tag{A.11}$$

==>

$$\nabla_{\boldsymbol{\theta}} \log(\psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})) = \frac{1}{\psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})} \frac{-\nu}{r(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})^2} \frac{[\nabla_{\boldsymbol{\theta}} \log(\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}))]\, \phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})}{q_k(\mathbf{z} \mid \mathbf{x}) p_{\mathbf{y}}(\mathbf{x})} \tag{A.12}$$

$$= \frac{1}{\psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})} \frac{-\nu}{r(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})} [\nabla_{\boldsymbol{\theta}} \log(\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}))] \tag{A.13}$$

$$= \frac{-\nu}{r(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) + \nu} [\nabla_{\boldsymbol{\theta}} \log(\phi(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}))] \tag{A.14}$$

Where in the final line we used the fact that $\psi_1(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) - 1 = \frac{\nu}{r(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})}$.

Now we find the derivative of the second term of $J^K_{\text{VNCE}}(\boldsymbol{\theta})$. Recalling that:

$$\psi_2(\mathbf{y}; \boldsymbol{\theta}) = 1 + \frac{1}{\nu} \mathbb{E}_{\mathbf{z} \sim q_k}[r(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta})], \tag{A.15}$$

we have:

$$\nabla_{\boldsymbol{\theta}} \log(\psi_2(\mathbf{y}; \boldsymbol{\theta})) = \frac{1}{\psi_2(\mathbf{y}; \boldsymbol{\theta})} \nabla_{\boldsymbol{\theta}} \psi_2(\mathbf{y}; \boldsymbol{\theta}) \tag{A.16}$$

$$= \frac{1}{\psi_2(\mathbf{y}; \boldsymbol{\theta})} \frac{1}{\nu} \mathbb{E}_{\mathbf{z} \sim q_k} \left[\frac{\nabla_{\boldsymbol{\theta}} \phi(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta})}{q_k(\mathbf{z} \mid \mathbf{y}) p_{\mathbf{y}}(\mathbf{y})} \right] \tag{A.17}$$

$$= \frac{1}{\nu} \frac{1}{\psi_2(\mathbf{y}; \boldsymbol{\theta})} \mathbb{E}_{\mathbf{z} \sim q_k} \left[r(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta})\, [\nabla_{\boldsymbol{\theta}} \log(\phi(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta}))] \right] \tag{A.18}$$

==>

$$\nabla_{\boldsymbol{\theta}}(J^k_{\text{VNCE}}(\boldsymbol{\theta})) = \mathbb{E}_{\mathbf{x}} \mathbb{E}_{\mathbf{z} \sim q_k} \frac{\nu}{r(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) + \nu} \nabla_{\boldsymbol{\theta}} E(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}) \tag{A.20}$$

$$- \mathbb{E}_{\mathbf{y}} \frac{1}{1 + \frac{1}{\nu} \mathbb{E}_{\mathbf{z} \sim q_k}[r(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta})]} \mathbb{E}_{\mathbf{z} \sim q_k} r(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} E(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta})$$

A.3 Experimental details for toy approximate inference problem

In section 4.4 we approximated a posterior $p(\mathbf{z} \mid \mathbf{x})$ with a variational distribution $q(\mathbf{z}|\mathbf{x};\boldsymbol{\alpha}) = \mathcal{N}(\mathbf{z};\boldsymbol{\mu}(\mathbf{x};\boldsymbol{\alpha}),\boldsymbol{\Sigma}(\mathbf{x};\boldsymbol{\alpha}))$, where $\boldsymbol{\Sigma}$ is diagonal, and $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are reparametrised by a single 2-layer feed-forward NN with weights $\boldsymbol{\alpha}$.

A.4 Univariate conditionals of a truncated Gaussian

As shown in Section 5.2, the truncated Gaussian can be written as

$$\phi(\mathbf{x};\boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}\mathbf{x}^T K \mathbf{x} + \mathbf{x}^T \mathbf{b}\right) \mathbb{I}(\mathbf{x} \in [0,+\infty]^d). \tag{A.21}$$

We can obtain the univariate conditional $\phi(x_j \mid \mathbf{x}_{\setminus j} = \mathbf{a};\boldsymbol{\theta})$, as follows. Assuming $\mathbf{x} \in [0,+\infty]^d$,

$$\begin{aligned}
\phi(x_j \mid \mathbf{x}_{\setminus j} = \mathbf{a};\boldsymbol{\theta}) &\propto \phi(x_j, \mathbf{x}_{\setminus j} = \mathbf{a};\boldsymbol{\theta}) && \text{(A.22)}\\
&\propto \exp\left(-\frac{1}{2}K_{j,j}x_j^2 + b_j x_j\right) \prod_{m \neq j} \exp\left(-K_{j,m} a_m x_j\right) && \text{(A.23)}\\
&\propto \exp\left(-\frac{1}{2}K_{j,j}x_j^2 + b_j x_j - K_{j,\setminus j}\mathbf{a}_{\setminus j} x_j\right) && \text{(A.24)}\\
&\propto \exp\left(-\frac{1}{2}x_j \bar{K}_j x_j + x_j \bar{b}_j\right), && \text{(A.26)}
\end{aligned}$$

where $\bar{K}_j = K_{j,j}$ and $\bar{b}_j = b_j - K_{j,\setminus j}\mathbf{a}_{\setminus j}$. Equation A.26 has the same form as Equation A.21, and so the conditional distribution is also a truncated normal.

It is possible to write down an expression for the *normalised* univariate truncated Gaussian [9]. By setting $\bar{\mu}_j = \bar{b}_j/\bar{K}_j$ and $\bar{\sigma}_j^2 = 1/\bar{K}_j$ we get,

$$p(x_j \mid \mathbf{x}_{\setminus j} = \boldsymbol{a};\boldsymbol{\theta}) = \frac{\psi(\frac{x_j - \bar{\mu}_j}{\bar{\sigma}_j})}{\left(1 - \Phi(\frac{-\bar{\mu}_j}{\bar{\sigma}_j})\right)\bar{\sigma}_j}, \tag{A.27}$$

where ψ is the pdf of a standard normal, Φ its cdf.