MLE/VAE for Nonlinear ICA

1 Introduction

1.1 Problem Setup

Suppose $X \in \mathbb{R}^n$ and $Z \in \mathbb{R}^m$ with $X = f(Z) + \varepsilon$, where $f : \mathbb{R}^m \to \mathbb{R}^n$ and $\varepsilon \sim N\left(0, \sigma^2 I_n\right)$. Assuming Z is normally distributed with independent marginals, this is equivalent to the following latent variable model (a special case of the well-known *nonlinear ICA* model):

$$\begin{aligned} Z &\sim N(0, I) \\ X \mid Z &\sim N\left(f(Z), \sigma^2 I\right). \end{aligned} \tag{1}$$

Let $\varphi(u; \mu, \Sigma)$ denote the density of a $N(\mu, \Sigma)$ random variable and $p_{\theta,\sigma^2}(x, z)$ denote the joint density under the model. It is easy to see that

$$\begin{aligned} p_{\theta,\sigma^2}(x, z) &= p_{\theta,\sigma^2}\left(x \mid z\right) p(z) = \varphi\left(x; f(z), \sigma^2 I\right) \varphi(z; 0, I) \\ L\left(\theta, \sigma^2; x\right) &= p_{\theta,\sigma^2}(x) = \int \varphi\left(x; f(z), \sigma^2 I\right) \varphi(z; 0, I) dz \end{aligned} \tag{2}$$

1.2 Objective Function

Now, suppose we let g_θ denote a family of deep neural network distributions parametrized by θ. To approximate the marginal density $p(x)$, we replace f with g_θ and try to find the choice of θ that maximizes the observed data likelihood. Given k observations $x^{(i)} \overset{i.i.d}{\sim} p(x)$, we wish to solve the following maximum likelihood problem:

$$\max_{\theta,\sigma^2} \underbrace{\sum_{i=1}^{k} \log \int \varphi\left(x^{(i)}; g_\theta(z), \sigma^2 I\right) \varphi(z; 0, I) dz}_{:=\ell(\theta,\sigma^2)} \tag{3}$$

2 Related Literature

It was widely believed in previous literature that directly optimizing the marginal likelihood in latent variable models is hard without making common simplifying assumptions about the marginal or posterior probabilities. In particular, we are interested in the intractability settings as described by (3), where

- The integral of the marginal likelihood $p_\theta(x) = \int p_{\boldsymbol{\theta}}(z) p_{\boldsymbol{\theta}}\left(x \mid z\right) dz$ is intractable (so we cannot evaluate or differentiate the marginal likelihood)
- The true posterior density $p_{\boldsymbol{\theta}}\left(z \mid x\right) = p_{\boldsymbol{\theta}}(x \mid z) p_{\boldsymbol{\theta}}(z) / p_{\boldsymbol{\theta}}(x)$ is intractable (so the EM algorithm cannot be used)
- The required integrals for any reasonable mean-field VB algorithm are also intractable.

The intractability condition is very common and can be easily established for moderately complicated likelihood functions $p_{\boldsymbol{\theta}}(x \mid z)$ (e.g. a neural network with a nonlinear hidden layer). Under the intractabilities, the direct MLE method is hard:

- P Dayan, et al. 1994 (4) established that the posterior probability of an explanation given d and θ is related to its energy by the equilibrium or Boltzmann distribution, which at a temperature of 1 gives

$$p_\theta(z \mid x) = \frac{p_\theta(z) \cdot p_\theta(x \mid z)}{\int p_{\boldsymbol{\theta}}(z') \cdot p_{\boldsymbol{\theta}}\left(x \mid z'\right) dz'} = \frac{e^{-E_z}}{\int e^{-E_{z'}} dz'}$$

  However, this posterior distribution is, in general, computationally intractable. It has exponentially many terms and cannot be factored into a product of simpler distributions.

In addition to the challenges proposed in previous literature, we have demonstrated in our own experiments, showing that directly optimizing likelihood 1 is computationally hard due to the following facts:

- The density $\varphi\left(x^{(i)}; g_\theta(z), \sigma^2 I\right)$ becomes vanishingly small when the dimension of the observed space is large, incurring numerical underflow when evaluating the likelihood and computing optimization. There is no simple work around of the issue, however, as the log-likelihood cannot be used when evaluating this numerical integration.
- Approximating the integral with numerical integration is challenging – which requires a large number of Monte-Carlo samples such that the integral can be evaluated to the precision desired. In contrast, (3) has found that the number of Monte-Carlo samples from the latent space can be set to one, as long as the minibatch size M is large enough – this avoids the computational burden of a large number of Monte-Carlo samples.

3 Methodologies

3.1 MLE with Gradient Descent

In this section, we directly solve the MLE problem by computing gradients of $\ell(\theta, \sigma^2)$ w.r.t θ and σ^2. This is, in general, intractable for arbitrary nonlinear ICA models but worst-case thinking does not apply to our special cases.

Gradient w.r.t θ

$$\begin{aligned}
\nabla_\theta \ell(\theta, \sigma^2) &= \sum_{i=1}^{k} \frac{1}{L\left(\theta, \sigma^2; x^{(i)}\right)} \int \nabla_\theta \varphi\left(x^{(i)}; g_\theta(z), \sigma^2 I\right) \varphi(z; 0, I) dz \\
&= \sum_{i=1}^{k} \frac{1}{L\left(\theta, \sigma^2; x^{(i)}\right)} \int \left(2\pi\sigma^2\right)^{-n/2} \nabla_\theta \exp\left(-\frac{\left\|x^{(i)} - g_\theta(z)\right\|_2^2}{2\sigma^2}\right) \varphi(z; 0, I) dz \\
&= \sum_{i=1}^{k} \frac{1}{L\left(\theta, \sigma^2; x^{(i)}\right)} \int \nabla_\theta \left(-\frac{\left\|x^{(i)} - g_\theta(z)\right\|_2^2}{2\sigma^2}\right) \varphi\left(x^{(i)}; g_\theta(z), \sigma^2 I\right) \varphi(z; 0, I) dz \\
&= \sum_{i=1}^{k} \frac{1}{L\left(\theta, \sigma^2; x^{(i)}\right)} \int \left[\frac{1}{\sigma^2} \cdot \nabla_\theta^T g_\theta(z) \left(x^{(i)} - g_\theta(z)\right)\right] \varphi\left(x^{(i)}; g_\theta(z), \sigma^2 I\right) \varphi(z; 0, I) dz
\end{aligned}$$

Gradient w.r.t σ^2

$$\begin{aligned}
\nabla_{\sigma^2}\ell(\theta,\sigma^2) &= \sum_{i=1}^{k}\frac{1}{L\left(\theta,\sigma^2;x^{(i)}\right)}\int \nabla_{\sigma^2}\varphi\left(x^{(i)};g_\theta(z),\sigma^2 I\right)\varphi(z;0,I)dz \\
&= \sum_{i=1}^{k}\frac{1}{L\left(\theta,\sigma^2;x^{(i)}\right)}\int \nabla_{\sigma^2}\left(2\pi\sigma^2\right)^{-n/2}\exp\left(-\frac{\left\|x^{(i)}-g_\theta(z)\right\|_2^2}{2\sigma^2}\right)\varphi(z;0,I)dz \\
&= \sum_{i=1}^{k}\frac{1}{L\left(\theta,\sigma^2;x^{(i)}\right)}\int \left[\frac{1}{2\sigma^2}\cdot\left(-n+\frac{\left\|x^{(i)}-g_\theta(z)\right\|_2^2}{\sigma^2}\right)\right]\varphi\left(x^{(i)};g_\theta(z),\sigma^2 I\right)\varphi(z;0,I)dz
\end{aligned}$$

From the two results above, we can iteratively update θ and σ^2 via gradient descent. The integrals can be approximated via numerical integration.

Algorithm 1 Direct MLE via Gradient Descent

- Initialise $\theta^{(0)}$ and ${\sigma^2}^{(0)}$ and set $t=0$
- Repeat until convergence
  - ▷ Compute the gradient $\nabla_\theta\ell\left(\theta^{(t)},{\sigma^2}^{(t)}\right)$ and update the parameters

$$\theta^{(t+1)} = \theta^{(t)} - \eta_1\nabla_\theta\ell\left(\theta^{(t)},{\sigma^2}^{(t)}\right)$$

  - ▷ Compute the gradient $\nabla_{\sigma^2}\ell\left(\theta^{(t+1)},{\sigma^2}^{(t)}\right)$ and update the parameters

$${\sigma^2}^{(t+1)} = {\sigma^2}^{(t)} - \eta_2\nabla_{\sigma^2}\ell\left(\theta^{(t+1)},{\sigma^2}^{(t)}\right)$$

  - ▷ Set $t \leftarrow t+1$

3.2 Variational Method

In this section, we consider variational inference and VAEs. We use the ELBO to obtain a lower bound on the likelihood $\ell(\theta,\sigma^2)$ and optimize the ELBO using SGD. The marginal likelihoods of individual datapoints can each be rewritten as

$$\log p_\theta\left(x^{(i)}\right) = D_{KL}\left(q_\phi\left(z^{(i)}\mid x^{(i)}\right)\|p_\theta\left(z^{(i)}\mid x^{(i)}\right)\right) + \mathcal{L}\left(\theta,\phi;x^{(i)}\right)$$

The term $\mathcal{L}\left(\theta,\phi;x^{(i)}\right)$ is called the evidence lower bound on the marginal likelihood of datapoint i and can be written as [1]

$$\begin{aligned}
\log p_\theta\left(x^{(i)}\right) \geq \mathcal{L}\left(\theta,\phi;x^{(i)}\right) &= \mathbb{E}_{q_\phi\left(z^{(i)}\mid x^{(i)}\right)}\left[-\log q_\phi\left(z^{(i)}\mid x^{(i)}\right)+\log p_\theta\left(x^{(i)},z^{(i)}\right)\right] \\
&= -D_{KL}\left(q_\phi\left(z^{(i)}\mid x^{(i)}\right)\|p_\theta\left(z^{(i)}\right)\right) + \mathbb{E}_{q_\phi\left(z^{(i)}\mid x^{(i)}\right)}\left[\log p_\theta\left(x^{(i)}\mid z^{(i)}\right)\right]
\end{aligned}$$

[1] An equivalent concept to ELBO is the variational free energy. The variational free energy in a latent variable model $p_\theta(x,z)$ is defined as

$$\mathcal{L}(\theta,q) = \mathbb{E}_{z\sim q}\left[-\log q(z)+\log p_\theta(x,z)\right],$$

where q is any probability density/mass function over the latent variables z. The first term is the Shannon entropy $H(q) = -\mathbb{E}_{z\sim q}\log q(z)$ of the variational distribution $q(z)$ and does not depend on θ. The second term is sometimes referred to as the energy.

We want to differentiate and optimize the lower bound $\mathcal{L}\left(\theta, \phi; x^{(i)}\right)$ w.r.t. both the variational parameters ϕ and generative parameters θ. The KL-divergence $D_{KL}\left(q_\phi\left(z^{(i)} \mid x^{(i)}\right) \| p_\theta\left(z^{(i)}\right)\right)$ can be integrated analytically, such that only the reconstruction error $\mathbb{E}_{q_\phi\left(z^{(i)} \mid x^{(i)}\right)}\left[\log p_\theta\left(x^{(i)} \mid z^{(i)}\right)\right]$ requires estimation by sampling. The *stochastic gradient variational bayes* (SGVB) estimator

$$\widetilde{\mathcal{L}}\left(\theta, \phi; x^{(i)}\right) = -D_{KL}\left(q_\phi\left(z^{(i)} \mid x^{(i)}\right) || p_\theta\left(z^{(i)}\right)\right) + \frac{1}{L}\sum_{l=1}^{L} \log p_\theta\left(x^{(i)} \mid z^{(i,l)}\right)$$

KL-divergence

When both the prior $p_\theta(z) = \mathcal{N}(0, I)$ and the posterior approximation $q_\phi\left(z^{(i)} \mid x^{(i)}\right)$ are Gaussian, the KL term that can be integrated analytically. Let J be the dimensionality of z. Let μ and σ denote the variational mean and std evaluated at datapoint i, and let μ_j and σ_j denote the j-th element of these vectors.

$$\begin{aligned} -D_{KL}\left(q_\phi(z \mid x) \| p_\theta(z)\right) &= \int q_\phi(z \mid x)\left(\log p_\theta(z) - \log q_\phi(z \mid x)\right) dz \\ &= \frac{1}{2}\sum_{j=1}^{J}\left(1 + \log\left(\left(\sigma_j\right)^2\right) - \left(\mu_j\right)^2 - \left(\sigma_j\right)^2\right) \end{aligned}$$

Reconstruction Error

In variational auto-encoders, neural networks are used as probabilistic encoders and decoders. For both the encoder and decoder, we use a MLP with Gaussian outputs. Let the decoder be a multivariate Gaussian with a diagonal covariance structure

$$\begin{aligned} \log p\left(x \mid z\right) &= \log \mathcal{N}\left(x; m, s^2 I\right) \\ \text{where } h &= h\left(z\right) \\ m &= W_1 h + b_1 \\ \log s^2 &= W_2 h + b_2 \end{aligned}$$

where $\{W_1, W_2, b_1, b_2\}$ are the weights and biases of the MLP (as part of θ) with $m \in R^n$ and $s^2 \in R$. The reconstruction error can be expanded as

$$\log p\left(x \mid z\right) = -\frac{n}{2} \cdot \log\left(2\pi s^2\right) - \frac{\|x - m\|_2^2}{2s^2}$$

4 Empirical Study

4.1 Experimental Setup

Data Generating Process

In the first part of our experiments, we fix $m = 2$ and $n = 20$. We fix the standard deviation $\sigma^2 = 1$ without learning it (it will help us compare the learned function $\hat{f}$ and the original f, hence the identifiability of the data generating model). For data generation, we use a single-layer MLP (which could, however, be piece-wise affine) followed by a non-linear activation. We have used four types of non-linear activation functions in our experiments: `ReLU`, `Sigmoid`, `Tanh`, and `Leaky ReLU`, among which `ReLU` and `Leaky ReLU` are piece-wise affine.

Below we visualize the sample distribution of the latent variable z and the generated x

- **Column:** Distribution of z, x with `ReLU` activation, x with `Sigmoid` activation, x with `Tanh` activation and x with `Leaky ReLU` activation
- **Row:** Distribution of z and x with $m = 1, n = 1$, $m = 1, n = 2$ and $m = 2, n = 2$

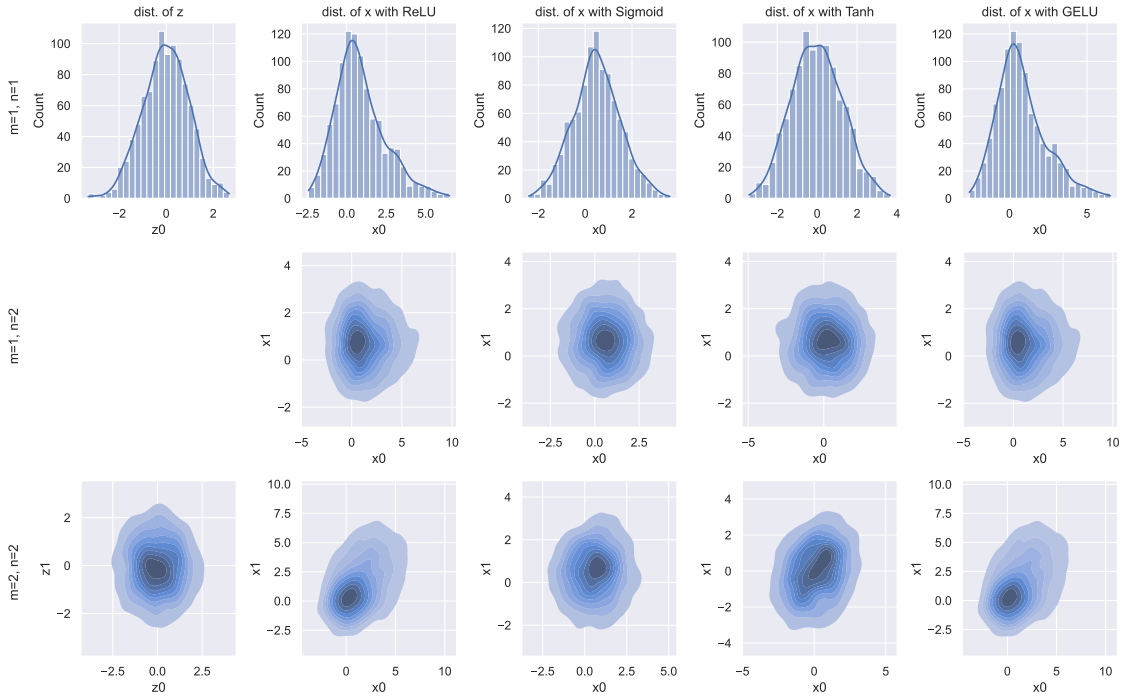

Figure 1: Visualization of the distribution of z and x with different specification of z, x and activation functions `ReLU`, `Sigmoid`, `Tanh`, and `Leaky ReLU`

Likelihood Evaluation

In this work, we have implemented two methods for evaluating the likelihood

- **Monte-Carlo Method:** We replace the integral over z using Monte-Carlo samples from the normal distribution. The Monte-Carlo estimator $\tilde{L}_{MC}$ for the likelihood is

$$\tilde{L}_{MC}\left(\theta, \sigma^2; x\right) = \frac{1}{B}\sum_{i=1}^{B} \varphi\left(x^{(i)}; f(z^{(i)}), \sigma^2 I\right) \quad \text{with} \quad z^{(i)} \overset{iid}{\sim} N(0, I)$$

- **Sparse Grid Method:** We create a grid with size $k \times k$ within the square $[-l, l]$, with k representing the granularity of the grid which we set as $k = 51, l = 2.5$. The sparse grid estimator $\tilde{L}_{SG}$ for the likelihood is

$$\tilde{L}_{SG}\left(\theta, \sigma^2; x\right) = \frac{1}{k^m}\sum_{i_1=1}^{k} \cdots \sum_{i_m=1}^{k} \varphi\left(x^{(i)}; f(z^{(i_1,\ldots i_m)}), \sigma^2 I\right) \quad \text{with} \quad z_j = \left(-1 + \frac{2j}{k}\right) l$$

Optimization

We have two implementations for optimization, the first of which makes use of the `torch.autograd` framework provided by `torch`, the second of which evaluates the gradient manually, as was outlined in Section 3. Each of the two optimization methods can be used together with the two methods

for likelihood evaluation, giving us four methods for evaluating the likelihood and performing optimization.

4.2 Evaluation Metrics

For evaluating the quality of the fitted models, we use data generated out-of-sample.

Orthogonal Procrustes Analysis

Given two identically sized matrices X and Y, orthogonal Procrustes analysis standardizes both such that:

- $\operatorname{tr}\left(AA^T\right) = 1$
- Both sets of points are centered around the origin

Procrustes then applies the optimal transform to the second matrix (including scaling/dilation, rotations, and reflections), so that the columns of X and Y match up as much as possible, that is $\|X - AY\|_F$ is minimized under the constraint $A^\top A = I$. The minimized objective function is called the *disparity score*, as a measure of the distance between the truth data matrix and the estimation upon the transformation with the matrix A.

Canonical Correlation Analysis (CCA)

An alternative and auxiliary evaluation metric we adopt is the canonical correlation analysis. In particular, we use the first sample canonical correlation to measure the disparity between the two data matrices.

Given random variables $X_1, \ldots, X_p$ and $Y_1, \ldots, Y_q$, we assemble them into two random vectors

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_p \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_q \end{bmatrix}$$

The canonical correlations can be found as follows: consider two linear combinations

$$U = \mathbf{a}^\top \mathbf{X}, \quad V = \mathbf{b}^\top \mathbf{Y}$$

with $\mathbf{a} \in \mathbb{R}^p$ and $\mathbf{b} \in \mathbb{R}^q$ constant vectors (note that U and V are random variables). The first pair of population canonical variables is defined by

$$\begin{aligned} (U_1, V_1)\big) &= \operatorname{argmax} \operatorname{Corr}(U, V) \\ &= \operatorname{argmax}\{\operatorname{Cov}(U, V) : \operatorname{Var}(U) = 1 = \operatorname{Var}(V)\} \end{aligned}$$

In general, the k^{th} pair of population canonical variables is defined by

$$\begin{aligned} (U_k, V_k) = \operatorname{argmax}\{ &\operatorname{Cov}(U, V) : \operatorname{Var}(U) = \operatorname{Var}(V) = 1, \\ &\operatorname{Cov}\left(U, U_j\right) = \operatorname{Cov}\left(V, V_j\right) = \operatorname{Cov}\left(U, V_j\right) = \operatorname{Cov}\left(V, U_j\right) = 0, j = 1, \ldots, k-1\Big\} \end{aligned}$$

for $k = 1, \ldots, p$. If we replace `argmax` by `max`, the corresponding maximal values are called the k^{th} population canonical correlation and denoted

$$\rho_k = \operatorname{Corr}\left(U_k, V_k\right)$$

Scaling of Disparity Score vs. CCA

In this subsection, we examine the scaling of disparity score and CCA against the dimension of the data matrix. In particular, we generate two data matrices from i.i.d normal distribution with increasing dimensions and examine the scaling of the dis-similarity measures against the dimension. The dis-similarity is expected to be large consistently, given the two matrices are generated independently from normal distributions.

In our simulations, we observe that the disparity score is consistently large under the Procrustes analysis. However, the first canonical correlation from CCA increases with the data dimension, an indication that the canonical correlation is a poor metric for comparing the dis-similarities between matrices across different dimensions.

With a sample size equal to 5000 and data dimension 2, we have the disparity score = 1.0 from Procrustes analysis and canonical correlation = 0.017 from CCA. Increasing the data dimension to 500, the disparity score is reduced to 0.93, and the canonical correlation is increased significantly to 0.594, a clear indication that the CCA is not a good metric for the evaluation of disparity.

4.3 Experimental Results

In our experiments, we visualize and compare the observed space x and the reconstruction $\hat{x}$ from VAE/MLE. If the dimension of the space is larger than 2D, we map the data $p\left(x \mid z\right)$ and the reconstruction $\widehat{p}\left(x \mid z\right)$ into a lower-dimensional 2D representation. We also visualize the learning curves (log-likelihood for MLE and the ELBO for VAE) against the number of epochs.

In addition to the observed space x, we have also obtained the latent representation $\widehat{p}\left(z \mid x\right)$ during VAE training and $p\left(z \mid x\right)$ from Monte-Carlo simulation, based on which the latent space representation can be compared. The latent representation $\widehat{p}\left(z \mid x\right)$ cannot be obtained for direct MLE, thus does not constitute a coherent measure of the quality of reconstruction that can be used for both MLE and VAE. We have thus decided to exclude it from our analysis in the following sections.

Experimental Results with MLE

The plots below summarize the data distribution in the observed space (in blue) and the corresponding reconstruction (in red) from MLE. From the plots, it is observed that MLE achieves very good reconstruction of the observed space for `ReLU` and `LeakyReLU` activation functions.

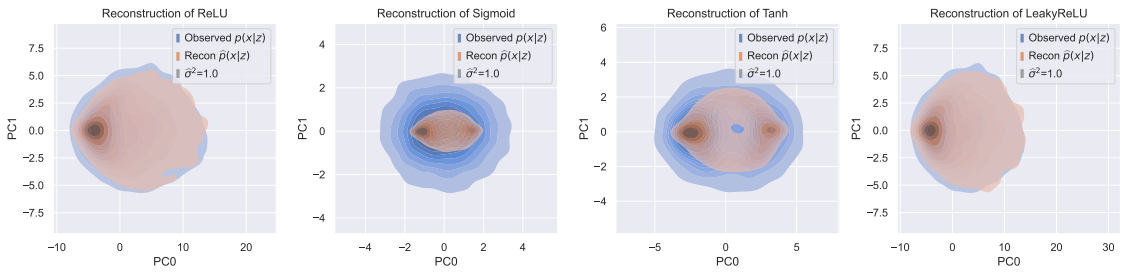

Figure 2: Visualization of the original observed space and the reconstruction with MLE

From the learning curve of MLE, we observe that the likelihood saturates very quickly after around 25 iterations. The likelihood on the training set and the validation set scale identically, suggesting

no evidence of overfitting/underfitting.

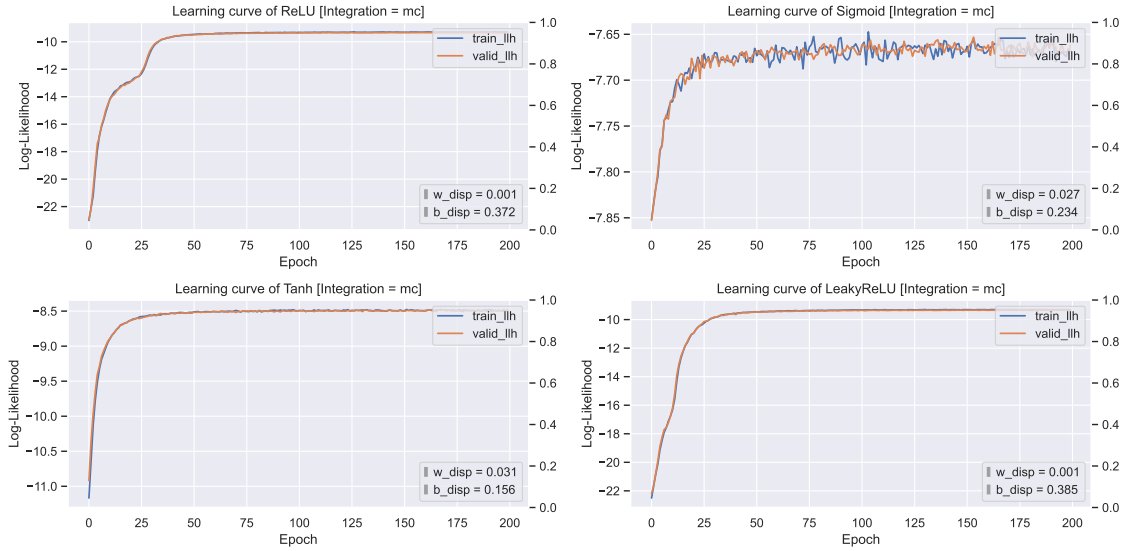

Figure 3: Learning curve for the training of MLE

Experimental Results with VAE

Using similar conventions, the plots below summarize the data distribution in the observed space (in blue) and the corresponding reconstruction (in red) from VAE. The quality of reconstruction is not as high as before (even for the piece-wise affine activation functions `ReLU` and `LeakyReLU`)

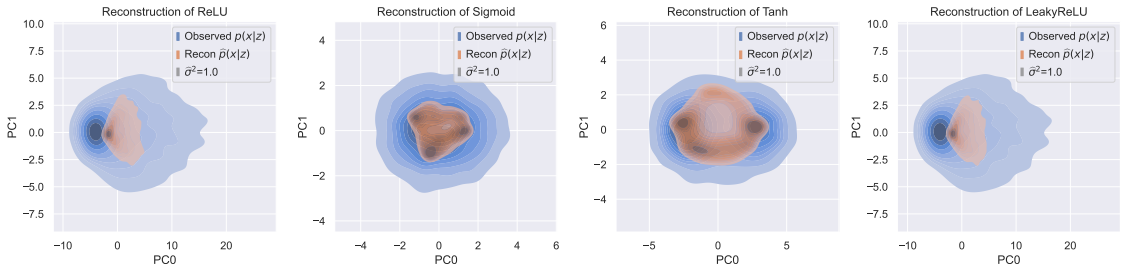

Figure 4: Visualization of the original observed space and the reconstruction with VAE

From the learning curve of VAE, some interesting observations can be made:

- The ELBO monotonically decreases and saturates very quickly after around 25 epochs of training
- The likelihood, on the other hand, does not demonstrate a monotonic pattern. Instead, for the Sigmoid activation function, the log-likelihood first increases and then decreases. The pattern is, however, not present in the other learning curves

Scaling of Procrustes disparity / CCA correlation with dimension

The CCA correlation has been observed to be a poor metric for evaluating the quality of reconstruction and should not be trusted when comparing the reconstruction vs. the original observed

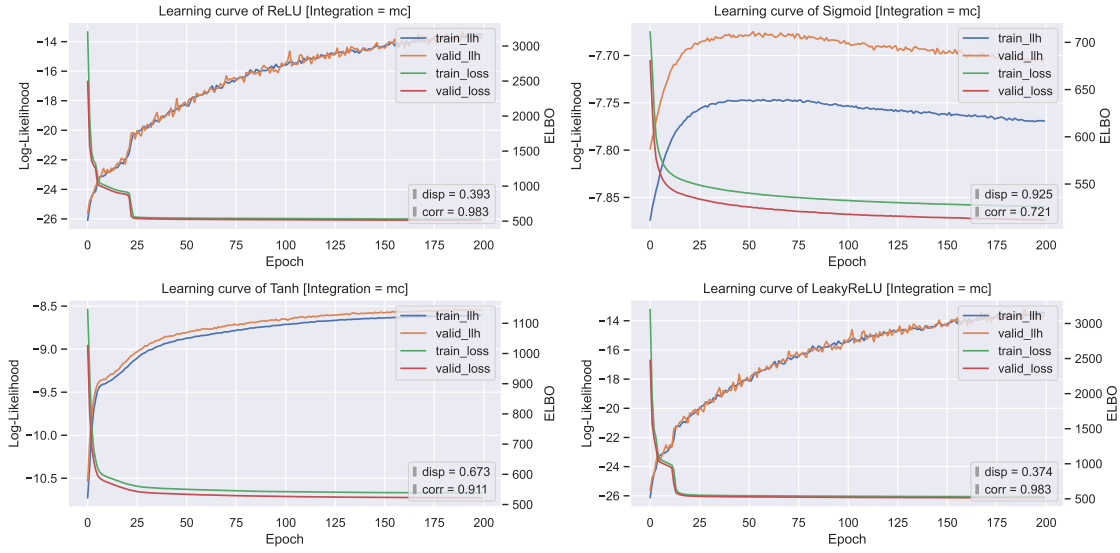

Figure 5: Learning curve for the training of VAE

data.

The Procrustes disparity score is observed to be consistent for increasing `n` with fixed `m=2` and is observed to be increasing with the latent dimension `m` while holding the observed dimension `n=75` fixed.

TODO: repeat the experiments and draw the confidence interval for the scaling. Also, do the same for MLE.

5 Evaluations

5.1 Numerical Integration

Monte Carlo

- Computationally efficient
- Work well in high dimensions (large 'm')
- High variance and require large enough samples

Sparse Grid

- Computationally less efficient
- Numerical underflow in high dimension
- Can evaluate exactly with a fine enough grid + complete coverage

5.2 Training and Inference

VAE

- **Pros:** Good computational scaling and efficiency (although training can be unstable for large m and n, heavily relying on initialization)

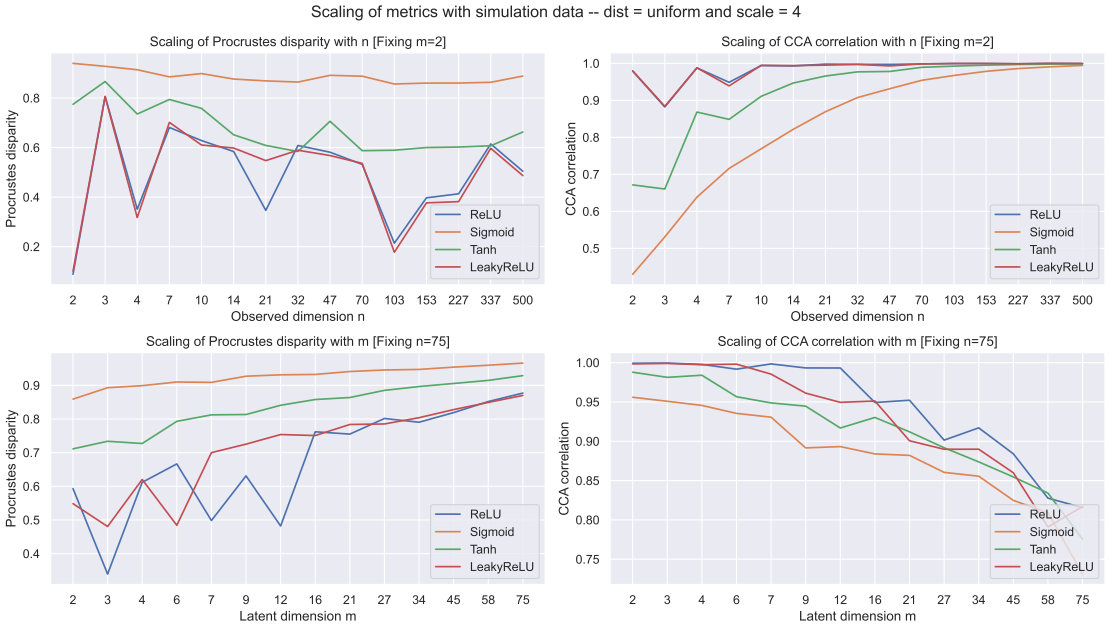

Figure 6: Scaling of Procrustes disparity and CCA correlation against the latent and the observed dimensions – VAE experiments

- **Cons:** As we optimize for the ELBO instead of directly optimizing for the likelihood, the quality of reconstruction is the best for variational distribution (rather than prior)
- **Notes:** Although the VAE method has better computational scaling, the scaling can be unstable (e.g. the optimizer produces invalid values for `m=100, n=100`)

Direct MLE

- **Pros:** The quality of reconstruction is better when MLE succeeds (need to verify more carefully with outputs from Procrustes analysis)
- **Cons:** The need for numerical integration with Monte Carlo sampling, which is computationally very inefficient
- **Cons:** The need for computing the likelihood (as opposed to the log-likelihood) — which induces poor computational scaling. The $\varphi\left(x^{(i)}; f(z^{(i)}), \sigma^2 I\right)$ term needs to be evaluated, which could run into numerical underflow when the dimension of $x^{(i)}$ is large. The computation cannot be easily scaled to `m=2, n=15`

$$\tilde{L}_{MC}\left(\theta, \sigma^2; x\right) = \frac{1}{B}\sum_{i=1}^{B} \varphi\left(x^{(i)}; f(z^{(i)}), \sigma^2 I\right) \quad \text{with} \quad z^{(i)} \overset{iid}{\sim} N(0, I)$$

5.3 Learning Curve

- **VAE:** The ELBO converges reasonably fast
- **VAE:** The LLH first deteriorates before improving again
- **VAE:** The LLH can deteriorate as we train for more iterations (e.g. Sigmoid)
- **MLE:** The LLH converges very fast

- **MLE:** Autograd does not work for `n=20` need to switch down to `n=10`

5.4 The Standard Deviation σ^2

- **VAE:** $\hat{\sigma}^2$ estimated using a network with VAE and can be data dependent
- **VAE:** $\hat{\sigma}^2$ underestimated with VAE
- **MLE:** $\hat{\sigma}^2$ estimated using an independent variable and not data dependent
- **MLE:** $\hat{\sigma}^2$ estimated correctly estimated with MLE

5.5 Observed Space

- Compare $p(x|z)$ with PCA
- Single latent sample per observation
- `m=2, n=20` has good reconstruction properties (in particular for `ReLU` and `GELU`)
- `m=2, n=2` may experience over-parametrization
- **VAE:** $p(x|z)$ is generated from the variational distribution
- **MLE:** $p(x|z)$ is generated from the prior

5.6 Latent Space

- MCMC for computing the posterior $p(z|x)$
- The posterior $p(z|x)$ resembles the observed space more than the latent
- Single latent sample per observation
- **VAE:** Latent $\hat{p}(z|x)$ available for VAE
- **VAE:** Latent variable z of interest or by-product of VAE structure
- **MLE:** Latent $\hat{p}(z|x)$ not available for MLE

References

[1] Khemakhem, Ilyes, et al. "Variational Autoencoders and Nonlinear ICA: A Unifying Framework." ArXiv.org, 21 Dec. 2020, arxiv.org/abs/1907.04809.

[2] Kingma, Diederik P., and Max Welling. "An Introduction to Variational Autoencoders." ArXiv.org, 11 Dec. 2019, https://arxiv.org/abs/1906.02691.

[3] Kingma, Diederik P, and Max Welling. "Auto-Encoding Variational Bayes." ArXiv.org, 1 May 2014, https://arxiv.org/abs/1312.6114.

[4] RS;, Dayan P;Hinton GE;Neal RM;Zemel. "The Helmholtz Machine." Neural Computation, U.S. National Library of Medicine, pubmed.ncbi.nlm.nih.gov/7584891/.

[5] Paisley, John, et al. "Variational Bayesian Inference with Stochastic Search." ArXiv.org, 27 June 2012, arxiv.org/abs/1206.6430.