2nd Symposium on Advances in Approximate Bayesian Inference, 2019 1–17

Characterizing and Avoiding Problematic Global Optima of Variational Autoencoders

Yaniv Yacoby[*] YANIVYACOBY@G.HARVARD.EDU
Weiwei Pan [†] WEIWEIPAN@G.HARVARD.EDU
Finale Doshi-Velez FINALE@SEAS.HARVARD.EDU
Harvard University, Cambridge, MA

Introduction Variational Auto-encoders (VAEs) are deep generative latent variable models consisting of two components: a generative model that captures a data distribution $p(x)$ by transforming a distribution $p(z)$ over latent space, and an inference model that infers likely latent codes for each data point (Kingma and Welling, 2013). Recent work shows that traditional training methods tend to yield solutions that violate modeling desiderata: (1) the learned generative model captures the observed data distribution but does so while ignoring the latent codes, resulting in codes that do not represent the data (e.g. van den Oord et al. (2017); Kim et al. (2018)); (2) the aggregate of the learned latent codes does not match the prior $p(z)$. This mismatch means that the learned generative model will be unable to generate realistic data with samples from $p(z)$(e.g. Makhzani et al. (2015); Tomczak and Welling (2017)).

In this paper, we demonstrate that both issues stem from the fact that the global optima of the VAE training objective often correspond to undesirable solutions. Our analysis builds on two observations: (1) the generative model is unidentifiable – there exist many generative models that explain the data equally well, each with different (and potentially unwanted) properties and (2) bias in the VAE objective – the VAE objective may prefer generative models that explain the data poorly but have posteriors that are easy to approximate. We present a novel inference method, LiBI, mitigating the problems identified in our analysis. On synthetic datasets, we show that LiBI can learn generative models that capture the data distribution and inference models that better satisfy modeling assumptions when traditional methods struggle to do so.

Background A VAE is comprised of a generative model and an inference model. Under the generative model, we posit that the observed data and the latent codes are jointly distributed as $p_\theta(x,z) = p_\theta(x|z)p(z)$. The likelihood $p_\theta(x|z)$ is defined by a neural network f with parameters θ and an output noise model $\epsilon \sim p(\epsilon)$ such that $x|z = f_\theta(z) + \epsilon$. Direct maximization of the expected observed data log-likelihood $\mathbb{E}_{p(x)}\left[\log \int_Z p_\theta(x,z)dz\right]$ over θ is intractable. Instead, we maximize the variational lower bound (ELBO),

$$\mathbb{E}_{p(x)}[\log p_\theta(x)] \geq \mathbb{E}_{p(x)}\left[\mathbb{E}_{q_{\eta(x)}(z)}\left[\log\left(\frac{p_\theta(x|z)p(z)}{q_{\eta(x)}(z)}\right)\right]\right] = \text{ELBO}(\theta,\eta), \tag{1}$$

where $q_{\eta(x)} \in Q$ is a variational distribution with parameters $\eta(x)$. Since the bound is tight when $q_{\eta(x)}(z) = p_\theta(z|x)$, we aim to infer $p_\theta(z|x)$. To speed up finding the variational parameters η for some new input x, we train a neural inference model g with parameters

[*] YY acknowledges support from NIH 5T32LM012411-04 and from IBM Research
[†] WP is supported by the Institute of Computational Sciences at Harvard University

ϕ such that $g_\phi(x) = \eta(x)$; we denote the variational distributions $g_\phi(x)$ by $q_\phi(z|x)$. Thus, maximization of the ELBO can be expressed (Zhao et al., 2017):

$$\text{argmin}_{\theta,\phi} - \text{ELBO}(\theta, \phi) = \text{argmin}_{\theta,\phi} (\underbrace{D_{\text{KL}}[p(x)||p_\theta(x)]}_{\text{MLE Objective}} + \underbrace{\mathbb{E}_{p(x)}\left[D_{\text{KL}}[q_\phi(z|x)||p_\theta(z|x)]\right]}_{\text{Posterior Matching (PM) Objective}}). \tag{2}$$

We call the first term the "MLE objective" (minimizing it maximizes $p_\theta(x)$), and the second term the "posterior matching (PM) objective" (it encourages variational posteriors to match posteriors of the generative model). We denote their sum by $L(\theta, \phi)$. Lastly, let $\phi_{\text{GT}} = \text{argmin}_\phi L(\theta_{\text{GT}}, \phi)$, where θ_{GT} is the data generating (ground truth) model.

1. A Framework for Understanding Issues with the VAE Objective

We demonstrate two general ways wherein global optima of the ELBO correspond to undesirable models. In the following, we fix our variational family to be mean-field Gaussian.

Case 1: Learning the Inference Model Compromises the Quality of the Generative Model. Suppose that the variational family does not contain the posteriors of the data-generating-model. Then, often, inference must trade-off between learning a generative model that explains the data well and one that has posteriors that are easy for the inference network to approximate. Thus, the global minima of the VAE objective can specify models that both fail to capture the data distribution and whose aggregated posterior fails to match the prior.

As demonstration, consider the following model (described fully in Appendix C.2):

$$x|z = \text{Cholesky}\left(AA^\intercal + B\right) z + \epsilon, \quad z \sim \mathcal{N}\left(0, I\right), \quad \epsilon \sim \mathcal{N}\left(0, I \cdot \sigma_\epsilon^2 - B\right) \tag{3}$$

with $\sigma_\epsilon^2 = 0.01$, $B = \left[\begin{smallmatrix} 0.006 & 0 \\ 0 & 0.006 \end{smallmatrix}\right]$ and $A = \left[\begin{smallmatrix} 0.75 & 0.25 \\ 1.5 & -1.0 \end{smallmatrix}\right]$ as the data generating model. Here, we fix B (which also fixes the covariance of the observation noise) and learn the parameter $\theta = A$. In this example, the ground-truth posteriors are non-diagonal Gaussians. Here, the VAE objective can achieve a lower loss by compromising the MLE objective in order to better satisfy the PM objective – i.e. the VAE objective will prefer a model that fails to capture the data distribution but has a diagonal Gaussian posterior over the ground-truth model. Figure 1C shows the data distribution of the ground truth model $\theta_{\text{GT}}, \phi_{\text{GT}}$ (with $L(\theta_{\text{GT}}, \phi_{\text{GT}}) = 0.532$) differs from the distribution of the learned model θ^*, ϕ^* in Figure 1D (with $L(\theta^*, \phi^*) = 0.196$). Moreover, since the learned model fails to capture the data distribution, its aggregated posterior fails to match the prior (see Figures 1E vs. 1F):

$$p(z) = \mathbb{E}_{p_{\text{data}}(x)}\left[p_{\theta_{\text{GT}}}(z|x)\right] \neq \mathbb{E}_{p_{\text{data}}(x)}\left[p_{\theta^*}(z|x)\right] \approx \mathbb{E}_{p_{\text{data}}(x)}\left[q_{\phi^*}(z|x)\right] \tag{4}$$

Even when we restrict the class of generative models to ones that fit the data well, the posterior matching objective will still select a model with a simple posterior. Unfortunately, the selected generative model may have undesirable properties like uninformative latent codes. As demonstration, consider the model from Equation 3 with the data generating model specified by: $\sigma_\epsilon^2 = 0.01$, $A = \left[\begin{smallmatrix} 0.75 & 0.25 \\ 1.5 & -1.0 \end{smallmatrix}\right]$, and B is some diagonal matrix with values in $[0, \sigma_\epsilon^2]$. In this case, we fix A and and learn the parameter $\theta = B$. Since *the observation noise covariance* $I \cdot \sigma_\epsilon^2 - B$ *changes with* B, the data marginal is fixed at $p_\theta(x) =$

$\mathcal{N}\left(0, AA^\intercal + I \cdot \sigma_\epsilon^2\right)$ for every B. Thus, for every θ, the MLE objective is 0. However, although every choice of θ explain the data equally well, the posterior matching objective (and hence the VAE objective) is minimized by θ's whose posteriors have the least amount of correlation. Figure 1A shows that $L(\theta, \phi)$ prefers high value in the upper diagonal of B and low value in the lower diagonal. Figure 1B shows the informativeness of the latent codes for the corresponding θ. We see that the data to latent code mutual information $I(X; Z)$ corresponding to the θ selected by $L(\theta, \phi)$ is not optimal. That is, even if the true data generating model produces highly informative latent codes, the VAE objective may select a model that produces uninformative latent codes.

Discussion The principles of our analysis extend to non-linear VAEs and complex variational families. In the VAE objective, the posterior matching objective acts like a regularizing term, biasing the learned generative models towards simple models with posteriors that are easy to approximate (with respect to the choice of variational family). Thus, joint training of the inference and generative models introduces unintended and undesirable optima, which would not appear when these models are learned separately.

Case 2: Learning the Inference Model Selects an Undesirable Generative Model. Even if the variational family is rich, the inference for the posterior can nonetheless bias the learning for the generative model. It is well known that the generative model is non-identifiable under the MLE objective – there are many models that minimize the MLE objective. To focus on the effects of non-identifiability, let us assume that the variational family is expressive enough that it contains the posteriors of multiple models that could have generated the data. Then the posterior matching objective is 0 since we can find parameters ϕ such that $q_\phi(z|x) = p_\theta(z|x)$ for any such θ. Consequently, $L(\theta, \phi)$ has multiple global minima corresponding to the multiple generative models that maximizes the date likelihood. Some of these models may not satisfy our desiderata; e.g., the latent codes have low mutual information with the data.

As demonstration, consider the following model (fully described in Appendix C.1):

$$x|z = \theta \cdot z + \epsilon, \quad z \sim \mathcal{N}(0, 1), \quad \epsilon \sim \mathcal{N}\left(0, \sigma_\epsilon^2 - \theta^2\right) \tag{5}$$

In this case, the mean-field variational family includes the posterior $p_\theta(z|x)$ for all θ, i.e. the posterior matching objective can be fully minimized. Furthermore, every $\theta \in [0, \sigma_\epsilon^2]$ yields the same data marginal, $p_\theta(x) = \mathcal{N}\left(0, \sigma_\epsilon^2\right)$, and thus minimizes the MLE objective. However, not all choice of θ are equivalent. Given θ, the mutual information between the learned latent codes and the data is $I_\theta(X; Z) = \text{Const} - \frac{1}{2}\log(\sigma_\epsilon^2 - \theta^2)$. Thus, the set of global minima of $L(\theta, \phi)$ contain many models that produce uninformative latent codes.

Discussion We've shown that posterior collapse can happen at *global* optima of the VAE objective and that, in these cases, collapse cannot always be mitigated by improving the inference model (as in He et al. (2019)) or by limiting the capacity of the generative model (as in Bowman et al. (2015); Gulrajani et al. (2016); Yang et al. (2017)).

2. LiBI: A New Inference Framework for VAEs

In Section 1, we showed that common problems with traditional VAE training stem from the non-identifiability of the likelihood and the bias of the VAE objective towards models with

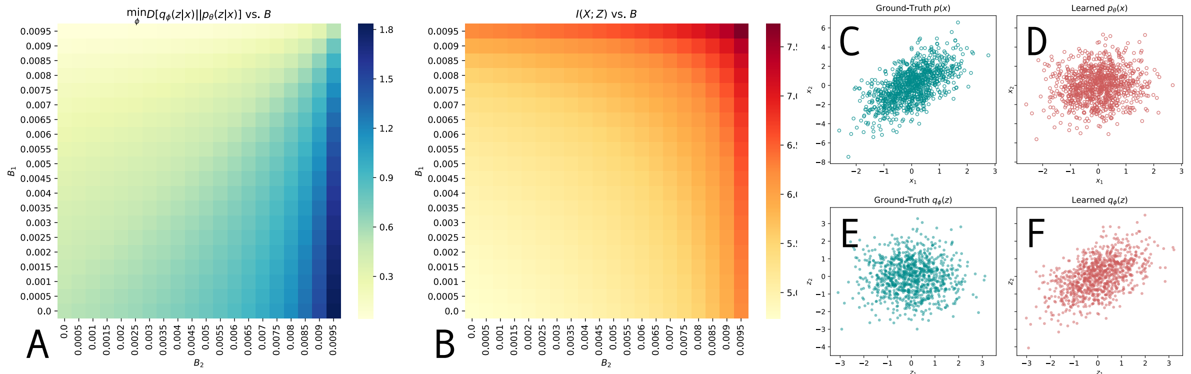

Figure 1: **A:** $\min_\phi D_{\mathrm{KL}}[q_\phi(z|x)||p_\theta(z|x)]$ vs. B for the model in Equation 3, showing that $L(\theta, \phi)$ prefers the B in the top-left corner. **B:** $I(X;Z)$ vs. B for the model in Equation 3, showing that the $I(X;Z)$ corresponding to the preferred B is not highest or lowest. **C & D:** true and learned data distributions for the model in Equation 3. **E & F:** true and learned aggregated posteriors for the model in Equation 3. Note: $q_\phi(z) \neq p(z)$.

simple posteriors, even if such models cannot capture the data distribution. We propose a novel inference method to *specifically* target these problems. To avoid the biasing effect of the PM objective on learning the generative model, we decouple the training of the generative and inference models – first we learn a generative model, then we learn an inference model while fixing the learned generative model (note that amortization allows for efficient posterior inference). To avoid undesirable global optima of the likelihood, we learn a generative model constrained by task-specific modeling desiderata. For instance, if informative latent codes are necessary for the task, the likelihood can be constrained so that the mutual information between the data and latent codes under θ is at least δ. While there are a number of works in literature that incorporate task-specific constraints to VAE training (e.g. Chen et al. (2016); Zhao et al. (2017, 2018); Liu et al. (2018)), adding these constraints to the VAE objective directly affects *both* the generative and the inference models, and, consequently, may introduce additional undesirable global optima. In our approach, added constraints only directly affects the generative model – i.e. the quality of inference cannot be compromised by the added constraints.

We call our training framework Likelihood Before Inference (LiBI), and propose *one possible instantiation* of this framework here.

Step 1: Learning the Generative Model We compute a tractable approximation to the MLE objective, constrained so that the likelihood satisfies task-specific modeling desiderata (such as high $I(X;Z)$) as needed.:

$$\mathrm{argmin}_\theta D_{\mathrm{KL}}[p(x)||p_\theta(x)] \quad \text{s.t} \quad c_i(\theta, X) < \epsilon_{c_i}, \forall i. \tag{6}$$

where each c_i is a constraint applied to the likelihood. We do this by computing joint maximum likelihood estimates for θ and z_n while additionally constraining the z_n's to have come from our assumed model (see Appendix D for a formal derivation of this approximation):

$$\begin{aligned}\mathrm{argmax}_{\theta,Z} \frac{1}{N}\sum_n \log p_\theta(x_n|z_n) \quad \text{s.t} \quad \mathrm{HZ}\left(\{z_n\}_{n=1}^N\right) < \epsilon_{\mathrm{HZ}}, \quad \left\|\Sigma\left(\{z_n\}_{n=1}^N\right) - I\right\|_2^2 < \epsilon_\Sigma, \\ \left\|\mu\left(\{z_n\}_{n=1}^N\right)\right\|_2^2 < \epsilon_\mu, c_i(\theta, X) < \epsilon_{c_i}, \forall i.\end{aligned} \tag{7}$$

where HZ$(\cdot)$ is the Henze-Zirkler test statistic for Gaussianity, $\mu(\cdot), \Sigma(\cdot)$ represent the empirical mean and covariance, and the z_n's are amortized using a neural network $z_n = h(x_n; \varphi)$ parametrized by φ. These constraints encourage the generative model to capture $p(x)$ given $p(z)$, i.e. the aggregated posterior under this model will match the prior $p(z)$.

Step 2: Learning the Inference Model Given the θ learned in Step 1, we learn ϕ to compute approximate posteriors $q_\phi(z|x)$: $\text{argmin}_\phi \mathbb{E}_{p(x)}[D_{\text{KL}}[q_\phi(z|x)||p_\theta(z|x)]]$. We note that ϕ, too, will satisfy our modeling assumptions, since with a fixed θ, the model non-identifiability we describe in Section 1 is no longer present.

Step 3: Reinitialize Inference for the Generative Model We repeat the process, initializing $h(x_n; \varphi) = \mu(x_n; \phi)$, where $\mu(x_n; \phi)$ is the mean of $q_\phi(z_n|x_n)$. This steps provides an intelligent random initialization allowing step 1 to learn a better quality model.

In theory, if the generative model and the inference models are learned perfectly in Steps 1 and 2, then Step 3 is obviated. In practice, we find that Step 3 improves the quality of the generative model and only a very small number of iterations is actually needed.

Discussion Using LiBI, we can now evaluate the quality of the generative model and the inference models independently. This is in contrast to traditional VAE inference, in which the ELBO entangles issues of modeling and issues of inference.

3. Experiments

On 4 synthetic data sets for which we know the data generating model, we compare LiBI with existing inference methods: VAE (Kingma and Welling, 2013), β-VAE (Higgins et al., 2017), β-VAE with annealing, Lagging inference networks (He et al., 2019). Across all datasets, LiBI learns generative models that better capture $p(x)$ (as quantified by log-likelihood and the Smooth k-NN test statistic (Djolonga and Krause, 2017)) and for which the aggregated posterior better matches the prior (see Appendix B).

	LinearJTEx		**CubicJTEx**		**Gaussian**		**Mobius**	
Method	**Test-LL**	**S-kNN**	**Test-LL**	**S-kNN**	**Test-LL**	**S-kNN**	**Test-LL**	**S-kNN**
VAE	-3.15 ± 0.04	1.62 ± 0.12	-5.85 ± 0.63	4.86 ± 1.82	6.73 ± 0.23	15.28 ± 7.98	1.88 ± 0.05	0.38 ± 0.20
β-VAE	-3.15 ± 0.04	1.62 ± 0.12	$\mathbf{-5.47 \pm 0.14}$	2.99 ± 1.48	7.65 ± 0.09	4.10 ± 1.22	$\mathbf{1.92 \pm 0.06}$	0.27 ± 0.14
β-VAE+Anneal	-3.15 ± 0.04	1.63 ± 0.12	-12.91 ± 11.51	2.86 ± 1.01	7.54 ± 0.14	5.86 ± 1.66	1.88 ± 0.05	0.37 ± 0.19
Lagging	-3.15 ± 0.04	1.62 ± 0.11	-30.64 ± 39.17	7.07 ± 1.24	6.94 ± 0.53	15.61 ± 8.26	1.90 ± 0.08	0.72 ± 0.78
LiBI (ours)	$\mathbf{-2.99 \pm 0.03}$	$\mathbf{0.06 \pm 0.05}$	-8.90 ± 3.98	$\mathbf{1.75 \pm 2.75}$	$\mathbf{7.85 \pm 0.05}$	$\mathbf{0.10 \pm 0.08}$	1.91 ± 0.05	$\mathbf{0.17 \pm 0.06}$

Table 1: Comparison of methods on synthetic data-sets. Test-LL is the average test log-likelihood (higher is better). S-kNN is the Smooth k-NN test statistic for similarity between $p(x)$ and $p_\theta(x)$ (smaller is better). Our method out-performs all other benchmarks. Note that on CubicJTEx Test-LL is unreliable (see Appendix E-Evaluation for details).

Conclusion In this paper, we show that commonly noted issues with VAE training are attributable to the fact that global optima of the VAE training objective often includes undesirable solutions. Based on our analysis, we propose a novel training procedure, LiBI, that avoid these undesirable optima while retaining the tractability of traditional VAE inference. On synthetic datasets, we show that LiBI able to learn generative models that capture the data distribution and inference models whose aggregated posterior matches the prior while traditional methods struggle to do so.

References

Matthias Bauer and Andriy Mnih. Resampled Priors for Variational Autoencoders. *arXiv:1810.11428 [cs, stat]*, October 2018. URL http://arxiv.org/abs/1810.11428. arXiv: 1810.11428.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. *arXiv e-prints*, art. arXiv:1511.06349, Nov 2015.

Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational Lossy Autoencoder. *arXiv e-prints*, art. arXiv:1611.02731, Nov 2016.

Adji B. Dieng, Yoon Kim, Alexander M. Rush, and David M. Blei. Avoiding Latent Variable Collapse With Generative Skip Models. *arXiv:1807.04863 [cs, stat]*, July 2018. URL http://arxiv.org/abs/1807.04863. arXiv: 1807.04863.

Josip Djolonga and Andreas Krause. Learning Implicit Generative Models Using Differentiable Graph Tests. *arXiv e-prints*, art. arXiv:1709.01006, Sep 2017.

Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. PixelVAE: A Latent Variable Model for Natural Images. *arXiv e-prints*, art. arXiv:1611.05013, Nov 2016.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. Lagging Inference Networks and Posterior Collapse in Variational Autoencoders. *arXiv:1901.05534 [cs, stat]*, January 2019. URL http://arxiv.org/abs/1901.05534. arXiv: 1901.05534.

Irina Higgins, Loc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *ICLR*, 2017.

Yoon Kim, Sam Wiseman, Andrew C. Miller, David Sontag, and Alexander M. Rush. Semi-Amortized Variational Autoencoders. *arXiv e-prints*, art. arXiv:1802.02550, Feb 2018.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv e-prints*, art. arXiv:1312.6114, Dec 2013.

Qi Liu, Miltiadis Allamanis, Marc Brockschmidt, and Alexander L. Gaunt. Constrained Graph Variational Autoencoders for Molecule Design. *arXiv e-prints*, art. arXiv:1805.09076, May 2018.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial Autoencoders. *arXiv:1511.05644 [cs]*, November 2015. URL http://arxiv.org/abs/1511.05644. arXiv: 1511.05644.

Ali Razavi, Aron van den Oord, Ben Poole, and Oriol Vinyals. Preventing Posterior Collapse with delta-VAEs. *arXiv:1901.03416 [cs, stat]*, January 2019. URL http://arxiv.org/abs/1901.03416. arXiv: 1901.03416.

Jakub M. Tomczak and Max Welling. VAE with a VampPrior. *arXiv:1705.07120 [cs, stat]*, May 2017. URL http://arxiv.org/abs/1705.07120. arXiv: 1705.07120.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural Discrete Representation Learning. *arXiv e-prints*, art. arXiv:1711.00937, Nov 2017.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. Improved Variational Autoencoders for Text Modeling using Dilated Convolutions. *arXiv e-prints*, art. arXiv:1702.08139, Feb 2017.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Information Maximizing Variational Autoencoders. *arXiv e-prints*, art. arXiv:1706.02262, Jun 2017.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. Towards Deeper Understanding of Variational Autoencoding Models. *arXiv:1702.08658 [cs, stat]*, February 2017. URL http://arxiv.org/abs/1702.08658. arXiv: 1702.08658.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. The Information Autoencoding Family: A Lagrangian Perspective on Latent Variable Generative Models. *arXiv e-prints*, art. arXiv:1806.06514, Jun 2018.

Appendix A. Related Work

Two common issues noted in VAE literature are posterior collapse and the mismatch between aggregated posterior and prior. Posterior collapse occurs when the posterior under both the generative model and approximate posterior learned by the inference model are equal the prior $p(z)$ (He et al., 2019). Surprisingly, under posterior collapse, the model is still able to generate samples from $p_{\text{data}}(x)$(e.g. Chen et al. (2016); Zhao et al. (2017)). This is often attributed to the fact the generative model is very powerful and is therefore able to maximize the log data marginal likelihood without the help of the auxiliary latent codes (van den Oord et al., 2017). Existing literature focuses on mitigating model collapse in one of the three ways: 1. modifying the optimization procedure to bias training way from collapse (He et al., 2019); 2. choosing variational families that make collapse less likely to occur (Razavi et al., 2019); 3. modifying the generative and inference model architecture to encourage more information sharing between the x's and the z's (Dieng et al., 2018). Although much of existing literature describes issue of posterior collapse and proposes methods to avoid it, less attention has been given to explaining why it occurs. He et al. (2019) conjecture that it occurs as a result of the joint training: since the likelihood changes over the course of training, it is incentivized to ignore the output of the inference network whose output in the early stages of training is not yet meaningful.

Mismatch between aggregated posterior and prior refers to the case when $q_\phi(z) \neq p(z)$, where

$$q_\phi(z) = \mathbb{E}_{p_{\text{data}}(x)}[q_\phi(z|x)] \approx \frac{1}{N} \sum_n q_\phi(z_n|x_n) \tag{8}$$

One might expect the two distributions to match because for any given likelihood θ, one should be able to recover the prior from the true posterior $p(z|x)$ as follows:

$$\mathbb{E}_{p(x)}[p(z|x)] = p(z) \tag{9}$$

An x produced by the generated model from a z that is likely under the prior but unlikely under the aggregate posterior may have "poor sample quality", since the the generative model is unlikely to have encountered such a z during training (Makhzani et al., 2015; Tomczak and Welling, 2017). Existing literature mitigate this issue by either increasing the flexibility of the prior to better fit the aggregate posterior (Tomczak and Welling, 2017; Bauer and Mnih, 2018) or developing a method to sample more robustly from the latent space (Zhao et al., 2017). Examples of the latter include training a second VAE to be able to generate z from u and then sampling from $p_\theta(x, z)$ using a Gibbs sampler (Zhao et al., 2017).

In this work, we provide a unifying analysis of both posterior collapse and mismatch, showing that both can occur as global optima of the VAE objective. Through our analysis, we also show that at these optima, neither issue can be reliably resolved by existing methods.

Appendix B. Qualitative Evaluation of the Learned Posterior and Aggregated Posterior

In Figures 2 and 3, we compare the posteriors learned by traditional VAE inference and by LiBI, respectively, on the synthetic dataset LinearJTEx. Here we demonstrate that traditional inference learns a generative model θ under which it is easy to approximate the corresponding posteriors. However, this comes at the cost of θ being unable to capture the data distribution. Figure 2 shows that the means of the ground-truth variational posteriors μ_{GT} (top-left) are able to approximate the means of the true posteriors (bottom-left). However, because in traditional inference the quality of θ can be compromised to ease the learning of ϕ, we see that the means of the posteriors of the learned θ (bottom-right) do not match the means of the posteriors of the ground truth θ_{GT}. As a result, the means of the learned variational posteriors (top-right) approximate the means of the posteriors under θ (bottom-right) instead of the means of the posterior under θ_{GT} (bottom-left). In contrast, Figures 3 shows that LiBI does not compromise the quality of θ to ease the task of inferring the posterior. Thus, the variational posteriors (top-middle) approximate the true posteriors under θ_{GT} (bottom-left). Figures 4 and 5 show the same trends on CubicJTEx.

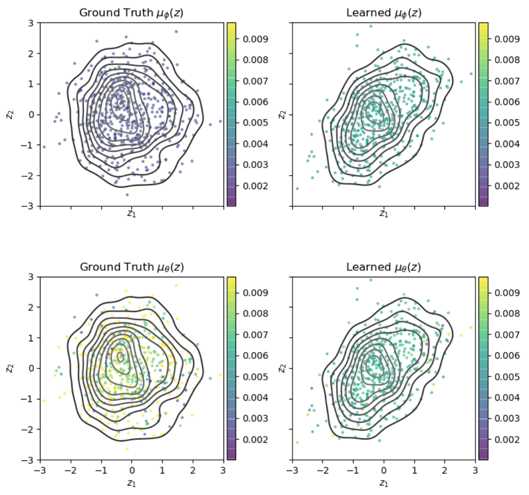

Figure 2: Visualization of Posterior learned by traditional VAE inference on LinearJTEx. Top-Left: Means of $q_{\phi_{\mathrm{GT}}}$, computed by minimizing the posterior matching objective given θ_{GT}. Bottom-Left: means of the true posterior, computed via an MC estimate of $p_{\theta_{\mathrm{GT}}}(z|x)$. Top-Right: learned mean of $q_{\phi_{\mathrm{GT}}}$. Bottom-Right: mean of posterior $p_\theta(z|x)$, computed via an MC estimate.

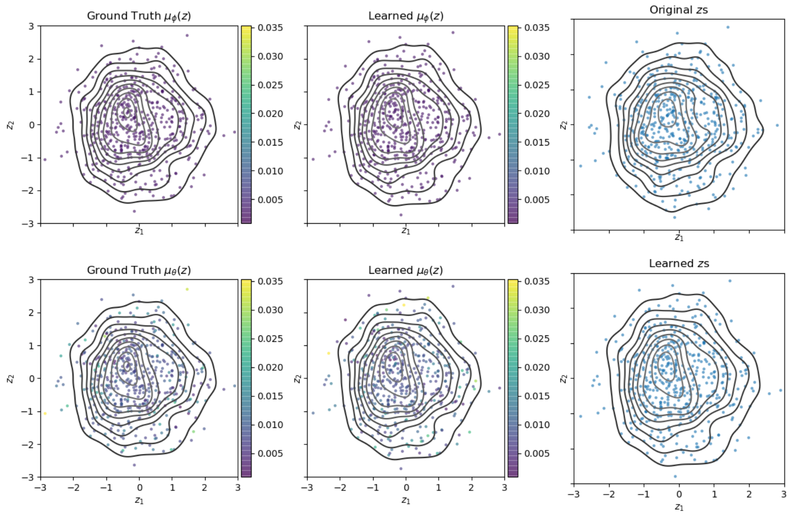

Figure 3: Visualization of Posterior learned by LiBI on LinearJTEx. Top-Left: Means of $q_{\phi_{\mathrm{GT}}}$, computed by minimizing the posterior matching objective given θ_{GT}. Bottom-Left: means of the true posterior, computed via an MC estimate of $p_{\theta_{\mathrm{GT}}}(z|x)$. Top-Middle: learned mean of $q_{\phi_{\mathrm{GT}}}$. Bottom-Middle: mean of posterior $p_\theta(z|x)$, computed via an MC estimate. Top-Right: original z's that generated the x's. Bottom-Right: z's learned via LiBI.

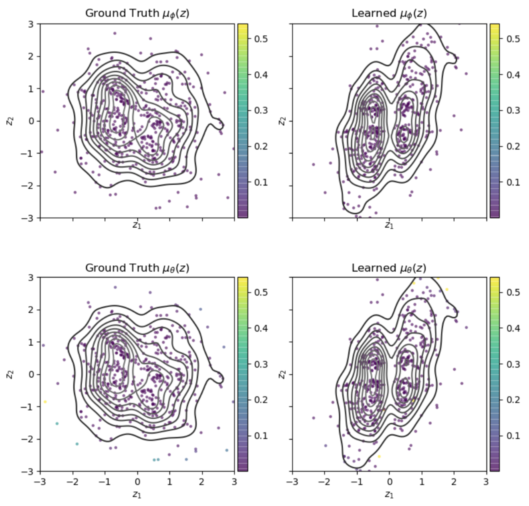

Figure 4: Visualization of Posterior learned by traditional VAE inference on CubicJTEx. Top-Left: Means of $q_{\phi_{\mathrm{GT}}}$, computed by minimizing the posterior matching objective given θ_{GT}. Bottom-Left: means of the true posterior, computed via an MC estimate of $p_{\theta_{\mathrm{GT}}}(z|x)$. Top-Right: learned mean of $q_{\phi_{\mathrm{GT}}}$. Bottom-Right: mean of posterior $p_\theta(z|x)$, computed via an MC estimate.

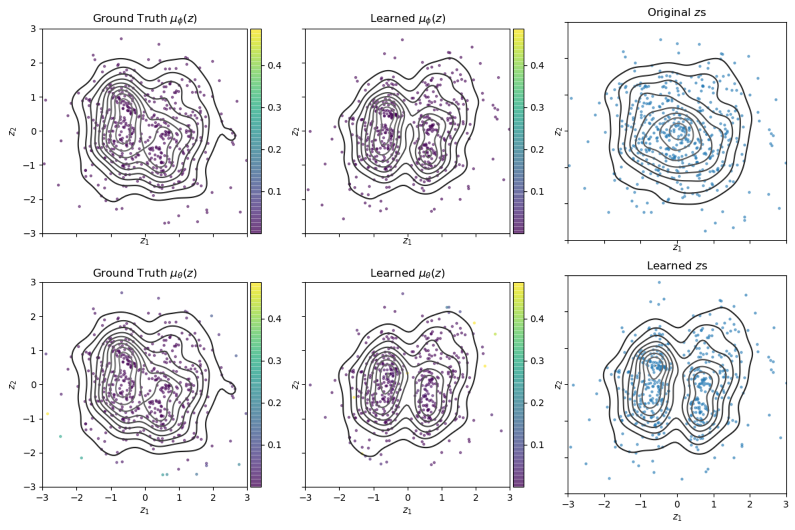

Figure 5: Visualization of Posterior learned by LiBI on CubicJTEx. Top-Left: Means of $q_{\phi_{\mathrm{GT}}}$, computed by minimizing the posterior matching objective given θ_{GT}. Bottom-Left: means of the true posterior, computed via an MC estimate of $p_{\theta_{\mathrm{GT}}}(z|x)$. Top-Middle: learned mean of $q_{\phi_{\mathrm{GT}}}$. Bottom-Middle: mean of posterior $p_\theta(z|x)$, computed via an MC estimate. Top-Right: original z's that generated the x's. Bottom-Right: z's learned via LiBI.

Appendix C. Pedagogical Examples

C.1. Case 1 Pedagogical Example

Assume the following generative process for the data:

$$\epsilon \sim \mathcal{N}\left(0, \sigma_\epsilon^2 - \theta^2\right) \tag{10}$$

$$z \sim \mathcal{N}(0, 1) \tag{11}$$

$$x|z = \theta \cdot z + \epsilon \tag{12}$$

For this generative process, $p_\theta(x) = \mathcal{N}\left(0, \sigma_\epsilon^2\right)$ for any value of θ such that $0 \leq \theta \leq \sigma_\epsilon^2$. Additionally, θ directly controls $I(X; Z)$ – when $\theta = 0$, we have that $I(X; Z)$; when $\theta = \sigma_\epsilon^2$, we have that $I(X; Z) = \infty$. To see this, we will compute $I_\theta(X; Z)$ directly (by computing $p_\theta(x, z)$ and $p(x)p(z)$):

$$p_\theta(x, z) = \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_\epsilon^2 & \theta \\ \theta & 1 \end{bmatrix}\right) \tag{13}$$

$$p_\theta(x)p(z) = \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_\epsilon^2 & 0 \\ 0 & 1 \end{bmatrix}\right) \tag{14}$$

As such, we can compute the mutual information between x and z as follows:

$$I_\theta(X; Z) = \frac{1}{2}\left[\log \frac{\sigma_\epsilon^2}{\sigma_\epsilon^2 - \theta^2} - 4\right] \tag{15}$$

For this model, the posterior $p_\theta(z|x)$, is:

$$p_\theta(z|x) = \mathcal{N}\left(\frac{\theta}{\sigma_\epsilon^2} \cdot x, \frac{\sigma_\epsilon^2 - \theta^2}{\sigma_\epsilon^2}\right) \tag{16}$$

Since this example is univariate, the mean-field Gaussian variational family will include the true posterior for any θ.

C.2. Case 2 Pedagogical Example

Assume the following generative process for the data:

$$\epsilon \sim \mathcal{N}\left(0, I \cdot \sigma_\epsilon^2 - B\right) \tag{17}$$

$$z \sim \mathcal{N}(0, I) \tag{18}$$

$$x|z = \text{Cholesky}\left(AA^\intercal + B\right) z + \epsilon \tag{19}$$

where B is a diagonal matrix with diagonal elements between 0 and σ_ϵ^2. For this generative process, $p_B(x) = \mathcal{N}\left(0, AA^\intercal + I \cdot \sigma_\epsilon^2\right)$ for all valid values of B. For this model, the complete data likelihood and marginals are,

$$p_B(x, z) = \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} AA^\intercal + I \cdot \sigma_\epsilon^2 & \text{Cholesky}\left(AA^\intercal + B\right) \\ \text{Cholesky}\left(AA^\intercal + B\right)^\intercal & I \end{bmatrix}\right) \tag{20}$$

$$p_B(x)p(z) = \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} AA^\intercal + I \cdot \sigma_\epsilon^2 & 0 \\ 0 & I \end{bmatrix}\right) \tag{21}$$

Therefore, $I_B(X;Z)$ can be computed as follows:

$$I_B(X;Z) = \frac{1}{2}\left[\log\det(AA^\intercal + I\cdot\sigma_\epsilon^2) - \sum_{i=1}^{K}\log(\sigma_\epsilon^2 - B_{ii})\right] \tag{22}$$

Lastly, the posterior for this model, $p_B(z|x)$, is a Gaussian with mean and covariance,

$$\mu_{z|x} = \Sigma_{z|x}\text{Cholesky}\left(AA^\intercal + B\right)^\intercal\left(I\cdot\sigma_\epsilon^2 - B\right)^{-1}x \tag{23}$$

$$\Sigma_{z|x} = \left(I + \text{Cholesky}\left(AA^\intercal + B\right)^\intercal\left(I\cdot\sigma_\epsilon^2 - B\right)^{-1}\text{Cholesky}\left(AA^\intercal + B\right)\right)^{-1} \tag{24}$$

For our choice of A, the mean-field Gaussian will not include the true posterior for this model. The best-fitting mean-field approximation to the true posterior can be computed as in Appendix C.3.

C.3. Best-Fitting Mean-Field Gaussian to Multivariate Gaussian

Let B be a diagonal matrix and let Σ be a full-covariance matrix.

$$\text{argmin}_B D_{\text{KL}}\left[\mathcal{N}(0,B)||\mathcal{N}(0,\Sigma)\right] = \text{argmin}_B\frac{1}{2}\left[\log\det\Sigma - \log\det B + \text{tr}(\Sigma^{-1}B) - K\right] \tag{25}$$

$$= \text{argmin}_B - \log\det B + \text{tr}(\Sigma^{-1}B) \tag{26}$$

$$= \text{argmin}_B\sum_{i=1}^{K} - \log B_{ii} + B_{ii}\Sigma_{ii}^{-1} \tag{27}$$

where each element in the above sum is independent and is minimized when $B_{ii} = \frac{1}{\Sigma_{ii}^{-1}}$, and where Σ_{ii}^{-1} is the ith diagonal entry of Σ^{-1}.

Appendix D. Derivation of LiBI

The LiBI Framework The LiBI framework is composed of two steps: (1) learning a high-quality likelihood capable of generating the observed data distribution, and (2) fixing the likelihood learned in Step 1, performing inference to learn the latent codes given the data. We emphasize that our framework is general, so one can use various existing methods for either step. For example, one can use a GAN for Step 1, and MCMC sampling for Step 2. In this section, we derive a tractable approximation to Step 1 that can be easily enhanced to include constraints for task-specific desiderata, and that is amenable to gradient-based optimization methods.

Tractable Approximation to the MLE Objective

$$\text{argmin}_\theta D_{\text{KL}}[p_{\text{data}}(x)||p_\theta(x)] = \text{argmin}_\theta - \mathbb{E}_{p_{\text{data}}(x)}\left[\log p_\theta(x)\right] \tag{28}$$

$$= \text{argmin}_\theta - \mathbb{E}_{p_{\text{data}}(x)}\left[\log\mathbb{E}_{p(z)}[p_\theta(x|z)]\right] \tag{29}$$

$$\approx \text{argmin}_\theta - \frac{1}{N}\sum_n\log\mathbb{E}_{p(z)}[p_\theta(x_n|z)] \tag{30}$$

$$\approx \text{argmin}_{\theta,Z} - \frac{1}{N}\sum_n\log p_\theta(x_n|z_n)p(z_n) \tag{31}$$

wherein Equation 31, we approximate $\mathbb{E}_{p(z)}[p_\theta(x_n|z)]$ with a single sample, z_n, that makes its corresponding x_n most likely (this is analogous to the Empirical Bayes EB MAP Type II estimates often used to tune prior hyper-parameters). This step, however, has a problem: it is biased towards learning z_n's close to 0. We will now demonstrate that this issue exists and is a result of non-identifiability in the MLE estimate with respect to $\theta, \{z_n\}_{n=1}^N$. We then provide a solution to this problem.

Characterization of Non-Identifiability in Tractable Approximation Consider the following: let $Z = \{z_n\}_{n=1}^N$ be the true z's and θ used to generate the observed data, $X = \{x_n\}_{n=1}^N$ in the following generative process:

$$z_n \sim p(z) = \mathcal{N}(0, I) \tag{32}$$

$$x_n|z_n \sim \mathcal{N}(f_\theta(z_n), \sigma_\epsilon^2 \cdot I) \tag{33}$$

Now, consider, an alternative $\widehat{Z} = \{\hat{z}_n\}_{n=1}^N$ and $\hat{\theta}$ such that,

$$\hat{z}_n = \frac{z_n}{c^2} \tag{34}$$

$$f_{\hat{\theta}}(\hat{z}) = f_\theta\left(c^2 \cdot \hat{z}\right) \tag{35}$$

yielding the following alternative generative process:

$$\hat{z}_n \sim p(\hat{z}) = \mathcal{N}\left(0, \frac{1}{c} \cdot I\right) \tag{36}$$

$$x_n|\hat{z}_n \sim \mathcal{N}(f_{\hat{\theta}}(\hat{z}_n), \sigma_\epsilon^2 \cdot I) \tag{37}$$

Under these generative processes, both the data marginals and the likelihoods are equal:

$$p_\theta(x) = p_{\hat{\theta}}(x) \tag{38}$$

$$p_\theta(x|z) = p_{\hat{\theta}}(x|\hat{z}) \tag{39}$$

However, since in our model we assumed the prior is fixed $p(z) = \mathcal{N}(0, I)$, the alternate parameters $\widehat{Z}, \hat{\theta}$ are preferred by the joint log-likelihood when $c > 1$,

$$\log p_{\hat{\theta}}(x_n|\hat{z}_n) + \log \mathcal{N}(\hat{z}_n|0, I) > \log p_\theta(x_n|z_n) + \log \mathcal{N}(z_n|0, I), \tag{40}$$

since $\log p_{\hat{\theta}}(x_n|\hat{z}_n) = \log p_\theta(x_n|z_n)$ by construction and $\log \mathcal{N}(\hat{z}_n|0, I) > \log \mathcal{N}(z_n|0, I)$ since the $\hat{z}_n$'s are closer to 0 when $c > 1$. This will cause our approximation from Equation 31 to prefer the model $\hat{\theta}$, which generates a different data distribution that the true data distribution:

$$\mathbb{E}_{p(z)}[p_{\hat{\theta}}(x|z)] \neq \mathbb{E}_{p(z)}[p_\theta(x|z)] \tag{41}$$

Identifying the Tractable Approximation using the Henze-Zirkler Test Statistic Returning to our approximation of the MLE objective in Equation 31, we can avoid this issue by constraining the z_n's to have come from the prior:

$$\text{argmin}_\theta D_{\text{KL}}[p_{\text{data}}(x)||p_\theta(x)] \approx \text{argmin}_{\theta, Z} -\frac{1}{N}\sum_n \log p_\theta(x_n|z_n) \quad \text{s.t} \quad z_n \sim p(z) \tag{42}$$

We do this by constraining the z_n's to be Gaussian using the Henze-Zirkler test for Gaussianity and by constraining the empirical mean and covariance of the z_n's to be that of the standard normal:

$$\begin{aligned}\text{argmin}_\theta D_{\text{KL}}[p_{\text{data}}(x)||p_\theta(x)] \approx \text{argmax}_{\theta,Z} \frac{1}{N}\sum_n \log p_\theta(x_n|z_n) \\ \text{s.t} \quad \text{HZ}\left(\{z_n\}_{n=1}^N\right) < \epsilon_{\text{HZ}}, \\ \left\|\Sigma\left(\{z_n\}_{n=1}^N\right) - I\right\|_2^2 < \epsilon_\Sigma, \\ \left\|\mu\left(\{z_n\}_{n=1}^N\right)\right\|_2^2 < \epsilon_\mu\end{aligned} \tag{43}$$

We hypothesize that if the likelihood function, f_θ, is "smooth" and well-behaved (that is, that it maps nearby z's to nearby x's), that our approximation of the likelihood will come close to the true one.

Using this framework, we first recover a high-quality likelihood (a likelihood that, unlike in the traditional VAE objective, is not compromised to match the approximate posterior). Our framework therefore naturally encourages this likelihood to satisfy modeling assumptions; that is, if we find a θ for which the x's are reconstructed accurately given Gaussian z's, the aggregated posterior under θ, $p_\theta(z)$, will match the prior $p(z)$. Given this likelihood, we can then learn a posterior that accurately approximates $p_\theta(z|x)$. We note that ϕ, too, will satisfy our modeling assumptions, since with a fixed θ, the model non-identifiability we describe is no longer present.

The LiBI Inference Method We incorporate the constraints in Equation 43 as smooth penalties into the Lagrangian in Equation 44. We additionally define $h(x_n;\varphi)$ to be a neural network parameterized by φ that, given x_n, returns the specific z_n that generated it. φ allows us to amortize Equation 44. We repeat the following steps R times:

1. Step 1:

$$\begin{aligned}\theta_t, \varphi_t = \text{argmin}_{\theta,\varphi} - \frac{1}{N}\sum_n \log p_\theta(x_n|h(x_n;\varphi)) \\ + \epsilon_{\text{HZ}} \exp\left(\text{HZ}\left(\{h(x_n;\varphi)\}_{n=1}^N\right)\right) \\ + \exp\left(\frac{\left\|\Sigma\left(\{h(x_n;\varphi)\}_{n=1}^N\right) - I\right\|_2^2}{\epsilon_\Sigma}\right) \\ + \exp\left(\frac{\left\|\mu\left(\{h(x_n;\varphi)\}_{n=1}^N\right)\right\|_2^2}{\epsilon_\mu}\right)\end{aligned} \tag{44}$$

2. Step 2:

$$\phi_t = \text{argmin}_\phi \frac{1}{N}\sum_n D_{\text{KL}}[q_\phi(z_n|x_n)||p_{\theta_t}(z_n|x_n)] \tag{45}$$

$$= \text{argmin}_\phi \frac{1}{N}\sum_n -\text{ELBO}(\theta_t, \phi) \tag{46}$$

3. Step 3: Initialize $h(x_n; \varphi_{t+1}) = \mu(x_n; \phi_t)$ and repeat, where $\mu(x_n; \phi_t)$ is the mean of the variational posterior.

While theoretically, given a sufficiently advanced optimizer, there is no need to repeat the procedure multiple times, we find that the optimization in Equation 44 is challenging and that re-initializing h using the means of the posterior provides a helpful perturbation out of local minima, while still remaining close to other good solutions. In practice, we also noticed that it is helpful to return the best θ_t (and its corresponding ϕ_t) across all repetitions.

Note: one conceptual difference between our method and traditional VAE inference is that in traditional VAE inference, ϕ is regarded as the "encoder", while in our method, we regard ϕ as the inference network and φ as the encoder.

Appendix E. Experiments

Synthetic Data We ran our method on four synthetic data-sets:

1. Linear Joint Training Example (LinearJTEx): We fix the generative model to be that in Equation 3, with $\sigma_\epsilon^2 = 0.01$, $B = \left[\begin{smallmatrix} 0.006 & 0 \\ 0 & 0.006 \end{smallmatrix}\right]$ and $A = \left[\begin{smallmatrix} 0.75 & 0.25 \\ 1.5 & -1.0 \end{smallmatrix}\right]$ as the ground truth parameters, and with $\theta = A$. We constrain Q to be the mean-field Gaussian variational family.

2. Cubic Joint Training Example (CubicJTEx): We fix the generative process to be that of Linear JTEx with one difference – we add a non-linearity to the likelihood function: $x|z = (\text{Cholesky}\,(AA^\intercal + B)\, z)^3 + \epsilon$, where the cubed-function is applied element-wise.

3. Gaussian: We use a linear likelihood function $x|z = z^\intercal A + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.000001)$ and $A = \left[\begin{smallmatrix} -0.7074 & 0.0995 & 0.0286 & 0.1240 \\ 0.7074 & 0.9948 & -0.9995 & 0.9920 \end{smallmatrix}\right]$.

4. Mobius: Let $m(z)$ be the Mobius Transform, $m(z) = \frac{a \cdot z + b}{c \cdot z + d}$, where z_1 and z_2 represent the real and imaginary parts of z, respectively, a, b, c, d are constants, and $m(z)$ is defined in terms of complex addition, multiplication and division. We set $a = [1, 0], b = [1, 4], c = [1, 0], d = [7, 4]$ and train a neural network f to map $z \sim p(z)$ to $m(z)$. We use this neural network approximation and the ground-truth function and use it to generate x: $x|z = f(z) + \epsilon$, where $\mathcal{N}(0, I \cdot \sigma_\epsilon^2$ and $\sigma_\epsilon^2 = 0.00001$.

For all data-sets, we constrain Q to be the mean-field Gaussian variational family. We also fix the hyper-parameters (σ_ϵ^2 and B) to be those of the true generative process. Lastly, on Linear JTEx and Gaussian, we can compute the posterior in closed-form. For the remaining data-sets, to get as close as possible to the ground-truth posterior, we fixed the likelihood to the ground truth and minimized the $L(\theta, \phi)$ with respect to ϕ only.

Training and Model Selection For each data-set type, we generated 5 data-sets, each consisting of 500 training, validation and test points. On each of the 10 data-sets, we ran 10 random restarts for each method and hyper-parameters (listed below). For each random-restart, we selected the learned model preferred by its own objective on the validation set. We averaged each method's performance across the 10 data-sets and present only the hyper-parameters on which the hyper-parameter choice results in highest average log-likelihood. Lastly, we trained each model for 30k epochs with a learning rate of 0.01.

Architecture:

- Generative Model, θ: For all models, we used the same architecture for the likelihood as the one of the ground-truth process.
- Inference Model, ϕ: We used linear encoders for LinearJTEx and Gaussian and a 1-hidden layer network with 50 hidden nodes ReLu activations for Mobius. Lastly, for CubicJTEx our encoder consisted of two hidden layers: the first with 4 hidden units, half with sigmoid activations and the other half with cube-root activations, and a second hidden layer with 20 hidden nodes with ReLu activations. We added the cube-root activations because of the difficulty inverting the cubic function in the generative process.
- Encoder, φ: We used the same architecture as the inference model on all data-sets.

Evaluation

- Average Test Log-Likelihood:

  $$\mathbb{E}_{p_{\text{data}}(x)}[\log p_\theta(x_n)] \approx \frac{1}{N}\sum_n \log \mathbb{E}_{p(z)}[p_\theta(x_n|z)] \tag{47}$$

  Since for our synthetic data, the likelihood is very peaky (that is, σ_ϵ^2 is small), to increase the sample efficiency of our estimates, we used importance sampling with the learned posterior as a proposal distribution:

  $$\mathbb{E}_{p_{\text{data}}(x)}[\log p_\theta(x_n)] \approx \frac{1}{N}\sum_n \log\left(\frac{1}{S}\sum_s \frac{p_\theta(x_n|z^{(s)})p(z^{(s)})}{q_\phi(z^{(s)}|x_n)}\right), \quad z^{(s)} \sim q_\phi(z_n|x_n) \tag{48}$$

  We inflated the variance of the proposal distribution by a factor of 2 to ensure our proposal has sufficient coverage. We used $S = 5000$ samples from the proposal. Even with importance sampling and a large number of samples, we found it difficulty estimating the log-likelihood on CubicJTEx.
- Smooth k-NN Two-Sample Test Statistic (Djolonga and Krause, 2017): lower values indicate that $p(x)$ matches $p_\theta(x)$. We computed the test statistics, comparing 100 randomly drawn samples generated from $p(x)$ to 100 randomly drawn samples generated from $p_\theta(x)$. We repeated this process 20000 times and reported the average.

Hyper-parameter Search For each data-set, we list below the hyper-parameter values we searched over:

1. LinearJTEx:
   - β-VAE with annealing: $\beta \in \{0.5, 1.0, 2.0, 5.0\}$
   - β-VAE without annealing: $\beta \in \{0.5, 2.0, 5.0\}$

- Lagging inference networks: $R \in \{40, 50, 60, 70\}$, where R here means we divide the total number of epochs into R equal segments. In each we train the inference network alone and then training the inference and generative networks jointly.
- LiBI: $\epsilon_{\text{HZ}} \in \{0.001, 1.0, 10.0, 20.0\}$, $\epsilon_{\Sigma} \in \{0.2, 0.5\}$, $\epsilon_{\mu} \in \{0.2, 0.5\}$, $\epsilon_{\mu} \in \{0.2, 0.5\}$, $R \in \{1, 6\}$.

2. CubicJTEx:
   - β-VAE with annealing: $\beta \in \{0.5, 1.0, 2.0, 5.0\}$
   - β-VAE without annealing: $\beta \in \{0.5, 2.0, 5.0\}$
   - Lagging inference networks: $R \in \{30, 40, 50, 60\}$, where R here means we divide the total number of epochs into R equal segments. In each we train the inference network alone and then training the inference and generative networks jointly.
   - LiBI: $\epsilon_{\text{HZ}} \in \{0.001, 1.0, 10.0, 20.0\}$, $\epsilon_{\Sigma} \in \{0.2, 0.5\}$, $\epsilon_{\mu} \in \{0.2, 0.5\}$, $R \in \{1, 6\}$.
3. Gaussian:
   - β-VAE with annealing: $\beta \in \{0.5, 1.0, 2.0, 5.0\}$
   - β-VAE without annealing: $\beta \in \{0.5, 2.0, 5.0\}$
   - Lagging inference networks: $R \in \{5, 10, 15, 20\}$, where R here means we divide the total number of epochs into R equal segments. In each we train the inference network alone and then training the inference and generative networks jointly.
   - LiBI: $\epsilon_{\text{HZ}} \in \{0.001, 1.0, 10.0, 20.0\}$, $\epsilon_{\Sigma} \in \{0.2, 0.5\}$, $\epsilon_{\mu} \in \{0.2, 0.5\}$, $R \in \{1, 6\}$.
4. Mobius:
   - β-VAE with annealing: $\beta \in \{0.5, 1.0, 2.0, 5.0\}$
   - β-VAE without annealing: $\beta \in \{0.5, 2.0, 5.0\}$
   - Lagging inference networks: $R \in \{60, 70, 80, 90\}$, where R here means we divide the total number of epochs into R equal segments. In each we train the inference network alone and then training the inference and generative networks jointly.
   - LiBI: $\epsilon_{\text{HZ}} \in \{1.0, 10.0, 20.0\}$, $\epsilon_{\Sigma} \in \{0.2, 0.5\}$, $\epsilon_{\mu} \in \{0.2, 0.5\}$, $R \in \{1, 6\}$.