WHERE TO DIFFUSE, HOW TO DIFFUSE AND HOW TO GET BACK: LEARNING IN MULTIVARIATE DIFFUSIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion-based generative models (DBGMs) perturb data to a target noise distribution and reverse this inference process to generate samples. The choice of inference diffusion affects both likelihoods and sample quality. Recent work in DBGMs has improved the inference process by extending it with auxiliary variables, leading to improved sample quality. While there are many such multivariate diffusions to explore, each new one requires significant model-specific analysis, hindering rapid prototyping and evaluation. In this work, we study Multivariate Diffusion Models (MDMs). For any number of auxiliary variables, we provide a recipe for maximizing a lower-bound on the MDMs likelihood without requiring any model-specific analysis. We then demonstrate how to parameterize the diffusion for a specified target noise distribution; these two points together enable optimizing the inference diffusion process. Optimizing the diffusion expands easy experimentation from just a few well-known processes to an automatic search over all linear diffusions. To demonstrate these ideas, we introduce two new specific diffusions as well as learn a diffusion process on the MNIST, CIFAR10, and IMAGENET32 datasets. We show MDMs improve bits-per-dims (BPDs), compared to the existing likelihood-trained VPSDE.

1 INTRODUCTION

Diffusion-based generative models (DBGMs) perturb data to a target noise distribution and reverse this inference process to generate samples. They have achieved impressive performance in image generation, editing, translation (Dhariwal & Nichol, 2021; Nichol & Dhariwal, 2021; Sasaki et al., 2021; Ho et al., 2022), conditional text-to-image tasks (Nichol et al., 2021; Ramesh et al., 2022; Saharia et al., 2022) and music and audio generation (Chen et al., 2020; Kong et al., 2020; Mittal et al., 2021). They are trained by maximizing a lower bound on the model log likelihood featuring an inference process, often interpreted as gradually "noising" the data (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020b).

The choice of inference process affects both likelihoods and sample quality. On different datasets and models, different inference processes work better; there is no universal best choice of inference, and the choice matters (Song et al., 2020b).

While some works have improved performance by experimenting with score model architectures (Ho et al., 2020; Kingma et al., 2021; Dhariwal & Nichol, 2021), Dockhorn et al. (2021) instead introduce the critically-damped langevin diffusion (CLD), showing that significant improvements in sample generation can be gained by carefully designing new processes. CLD pairs each data dimension with an auxiliary "velocity" variable and diffuses them jointly using second-order Langevin dynamics.

A natural question is, if introducing new diffusions results in dramatic performance gains, why are there only a handful of diffusions (variance-preserving stochastic differential equation (VPSDE), variance exploding (VE), CLD, sub-VPSDE) used in DBGMs? For instance, are there other auxiliary variable diffusions that would lead to improvements like CLD? This avenue seems promising as auxiliary variables have been used to improve other generative models and inferences, such as normalizing flows (Huang et al., 2020), neural ordinary differential equations (ODEs) (Dupont et al., 2019), hierarchical variational models (Ranganath et al., 2016), ladder variational autoencoder (Sønderby et al., 2016), and others.

Despite its success, CLD is evidence that each new process requires significant model-specific analysis. Deriving the evidence lower bound (ELBO) and training algorithm for diffusions is challenging (Huang et al., 2021; Kingma et al., 2021; Song et al., 2021) and is typically carried out in a case-by-case manner. Auxiliary variables seemingly complicate this process further. Computing conditionals of the inference process necessitates solving matrix Lyupanov equations (section 3.3). Deriving the inference stationary distribution—which helps the model and inference match—can be intractable. These challenges limit rapid prototyping and evaluation of new inference processes.

Concretely, to train a diffusion model one requires:

(R1): to select an inference and model process pair such that the inference process converges to the model prior

(R2): derive the ELBO for this pair

(R3): estimate the ELBO and its gradients by deriving and computing the inference process' transition kernel

In this work, we introduce Multivariate Diffusion Models (MDMs). MDMs are diffusion-based generative models trained with auxiliary variables. We provide a recipe for training MDMs beyond specific instantiations–like VPSDE and CLD—to all linear inference processes that have a stationary distribution, with any number of auxiliary variables.

First, we bring results from gradient-based MCMC (Ma et al., 2015) into diffusion modeling to construct MDMs that converge to a chosen model prior (**R1**); this tightens the ELBO. The parameterization spans all inference processes with a given stationary distribution. Secondly, for any number of auxiliary variables, we derive the MDM ELBO (**R2**). Finally, we show that the transition kernel of linear MDMs, neccesary for the ELBO, can be computed automatically and generically, and scaled to higher-order auxiliary systems, without incurring significant computational cost (**R3**).

With these tools, we explore a variety of new inference processes for diffusion-based generative models. We then note that the ELBO has no requirement for fixed inference; the automatic transition kernels and fixed stationary distributions facilitate directly learning the inference process to maximize the MDM ELBO. This learning turns diffusion model training into a search not only over score models but also over inference processes, at no extra derivational cost.

Contributions. In summary, our contributions are

1. Deriving ELBOs for training multivariate diffusion models (MDMs) with auxiliary variables.
2. Showing that the diffusion transition covariance does not need to be manually derived for each new diffusion. We instead demonstrate that a matrix factorization technique, previously unused in diffusion models, can automatically compute the covariance analytically for any linear MDM.
3. Using results from gradient-based Markov chain Monte Carlo (MCMC) to construct MDMs that exclude inference processes with poor likelihoods (i.e., a complete parameterization of inference processes whose stationary distribution matches the model prior).
4. Combining the above into an algorithm called Automatic Multivariate Diffusion Training (AMDT) that enables estimation without diffusion-specific derivations. AMDT enables training score models for *any* diffusion, including optimizing the diffusion and score jointly.

To demonstrate these ideas, we develop MDMs with two specified diffusions and learned multivariate diffusions. The specific diffusions are accelerated Langevin diffusion (ALDA), introduced in (Mou et al., 2019) as a higher-order scheme for gradient-based MCMC, and an alteration, modified accelerated Langevin diffusion (MALDA). Previously, using these diffusions for generative modeling would require significant model-specific analysis, but with our contributions, we can train without any derivations.

We train these models on the MNIST, IMAGENET32 and CIFAR10 datasets. Using MALDA and the learned diffusions, we achieve impressive bits-per-dim bounds, compared to the existing likelihood-trained VPSDE and CLD on the CIFAR-10, IMAGENET32 and MNIST datasets.

Empirical contributions. We show that

1. Existing diffusions such as ALDA, developed for MCMC sampling, can be adapted for generative modeling using the proposed algorithm AMDT.
2. Trying new diffusions, like MALDA, and learning diffusions using AMDT, is easy
3. For a given score model, MDMs improve bits-per-dim (BPD) compared to a univariate diffusion (VPSDE) on a number of benchmark datasets.
4. Using the MDM ELBO, there are new and existing MDMs that improve performance in BPD as much as a 3 fold increase in score model size for a fixed univariate diffusion.

2 SETUP

We present diffusions by starting with the generative model and then describing its likelihood lower bound (Sohl-Dickstein et al., 2015; Huang et al., 2021; Kingma et al., 2021). Diffusions sample from a model prior $\mathbf{z}_0 \sim \pi_\theta$ and then evolve a continuous-time stochastic process $\mathbf{z}_t \in \mathbb{R}^d$ according to:

$$d\mathbf{z} = h_\theta(\mathbf{z}, t)dt + \beta_\theta(t)d\mathbf{B}_t, \quad t \in [0, T] \tag{1}$$

where $\mathbf{B}_t$ is a d-dimensionsal Brownian motion. The model is trained so that $\mathbf{z}_T$ approximates the data $\mathbf{x} \sim q_{\text{data}}$.[1] Maximum likelihood training of diffusion models is intractable (Huang et al., 2021; Song et al., 2021; Kingma et al., 2021). Instead, they are trained using a variational lower bound on $\log p_\theta(\mathbf{z}_T = x)$. For this bound, an inference process $q_\phi(\mathbf{y}_s|\mathbf{x} = x)$ is specified for each time s:[2]

$$d\mathbf{y} = f_\phi(\mathbf{y}, s)ds + g_\phi(s)d\widehat{\mathbf{B}}_s, \quad s \in [0, T] \tag{2}$$

where $\widehat{\mathbf{B}}_s$ is another Brownian motion independent of $\mathbf{B}_t$. The inference process is usually taken to be specified rather than learned, and chosen to be i.i.d. for each y_{tj} conditional on each x_j. This leads to the interpretation of the y_{tj} as noisy versions of features x_j (Ho et al., 2020). While the diffusion ELBO is challenging to derive in general, when h_θ and β_θ are set to a particular choice $h_\theta(\cdot, t) = g^2(T-t)s_\theta(\cdot, T-t) - f(\cdot, T-t)$ and $\beta(t) = g(T-t)$ for a model s_θ, Huang et al. (2021); Song et al. (2021) show that the ELBO can be written as:

$$\log p_\theta(x) \geq \mathcal{L}^{\text{ism}}(x) = \mathbb{E}_{q_\phi(\mathbf{y}|x)}\left[\log \pi_\theta(\mathbf{y}_T) + \int_0^T -\frac{1}{2}\|s_\theta\|^2_{g^2_\phi} - \nabla \cdot (g^2_\phi s_\theta - f_\phi)ds\right], \tag{3}$$

where f_ϕ, g_ϕ, s_θ are evaluated at $(\mathbf{y}_s, s)$, and $\|\mathbf{x}\|^2_{\mathbf{A}} = \mathbf{x}^\top \mathbf{A}\mathbf{x}$. This features the Implicit Score Matching (ISM) loss (Song et al., 2020a). Equation (3) can also be written with the ISM replaced with the Denoising Score Matching (DSM) loss (Vincent, 2011; Song et al., 2020b), shown in appendix E.1.

3 A RECIPE FOR MULTIVARIATE DIFFUSION MODELS

As has been shown in prior work (Song et al., 2021; Dockhorn et al., 2021), the choice of diffusion matters. Drawing on principles from previous generative models (e.g. from normalizing flows and variational autoencoders (VAEs), mentioned in section 6), we can consider a wide class of diffusion inference processes by constructing them using auxiliary variables.

We now describe a method to design inference processes with auxiliary variables. At first glance, training such diffusions can seem challenging. First, one needs an ELBO that includes auxiliary variables. This ELBO will require sampling from the transition kernel, and setting the model prior to the inference stationary distribution. But doing such diffusion-specific analysis manually, is challenging and hinders rapid prototyping.

In this section we show how to address these challenges and introduce an algorithm, AMDT, to simplify and automate modeling with MDMs. AMDT can be used to train new and existing diffusions, including those with auxiliary variables, and including those that learn the inference process.

[1] Following Huang et al. (2021); Dockhorn et al. (2021) we integrate all processes in forward time 0 to T.

[2] We use $\mathbf{y}$ as the inference variable over the same space as the model's $\mathbf{z}$.

3.1 MULTIVARIATE MODEL AND INFERENCE

For the jth data coordinate, MDMs pair each $\mathbf{z}_{tj} \in \mathbb{R}$ with a $K-1$ dimensional vector of auxiliary variables $\mathbf{v}_{tj} \in \mathbb{R}^{K-1}$ and diffuse in the extended space:

$$\mathbf{u}_0 \sim \pi_\theta, \qquad d\mathbf{u} = h_\theta(\mathbf{u}_t = \begin{bmatrix}\mathbf{z}_t \\ \mathbf{v}_t\end{bmatrix}, t)dt + \beta_\theta(t)d\mathbf{B}_t. \tag{4}$$

MDMs model the data $\mathbf{x}$ with $\mathbf{z}_T$, a coordinate in $\mathbf{u}_T \sim p_\theta$. For the jth feature $\mathbf{x}_j$, each $\mathbf{u}_{tj} \in \mathbb{R}^K$ now consists of a primary "data" dimension (superscripts z) and auxiliary variables (superscripts v). Therefore $\mathbf{u} \in \mathbb{R}^{dK}$. We extend the drift coefficient h_θ from a function in $\mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}^d$ to the extended space $\mathbb{R}^{dK} \times \mathbb{R}_+ \to \mathbb{R}^{dK}$. We likewise extend the diffusion coefficient to a matrix β_θ acting on brownian motion $\mathbf{B}_t \in \mathbb{R}^{dK}$.

Because the MDM model is over the extended space, the inference distribution $\mathbf{y}$ must be too. We then set $q(\mathbf{y}_0^v|\mathbf{y}_0^z = x)$ to any chosen initial distribution, e.g. $\mathcal{N}(\mathbf{0}, \mathbf{I})$ and discuss this choice in section 4. Then $\mathbf{y}_s$ evolves according to an *inference process*,

$$d\mathbf{y} = f_\phi(\mathbf{y}, s)ds + g_\phi(s)d\widehat{\mathbf{B}}_s, \tag{5}$$

where the inference drift and diffusion coefficients f_ϕ, g_ϕ are now over the extended space $\mathbf{y} = [\mathbf{y}^z, \mathbf{y}^v]$. The function f_ϕ lets the z and v coordinates of $\mathbf{y}_{tj}$ interact in the inference process.

3.2 ELBO FOR MDMS

We now show how to train MDMs to optimize a lower bound on the log likelihood of the data. Like in the unvariate case, we parameterize $h_\theta(\cdot, t) = g^2(T-t)s_\theta(\cdot, T-t) - f(\cdot, T-t)$ and $\beta_\theta(t) = g(T-t)$.

Theorem 1. *The* MDM *log marginal likelihood of the data can be lower-bounded by:*

$$\begin{aligned}\log p_\theta(x) &\geq \mathbb{E}_{q_\phi(\mathbf{y}|x)}\left[\underbrace{\log \pi_\theta(\mathbf{y}_T)}_{\ell_T} - \int_0^T \frac{1}{2}\|s_\theta\|^2_{g_\phi^2} + \nabla \cdot (g_\phi^2 s_\theta - f_\phi)ds - \underbrace{\log q_\phi(\mathbf{y}_0^v|x)}_{\ell_q}\right] \quad (\mathcal{L}^{mism}) \\ &= \mathbb{E}_{q_\phi(\mathbf{y}|x)}\left[\ell_T + \int_0^T \frac{1}{2}\|s_q\|^2_{g_\phi} - \frac{1}{2}\|s_\theta - s_q\|^2_{g_\phi} + (\nabla \cdot f_\phi)ds - \ell_q\right] \quad (\mathcal{L}^{mdsm}).\end{aligned} \tag{6}$$

where divergences and gradients are taken with respect to $\mathbf{y}_s$ *and* $s_q = \nabla_{\mathbf{y}_s} \log q(\mathbf{y}_s|x)$.

Proof. The proof for the first bound is in appendix E. In short, we introduce auxiliary variables, apply Theorem 1 of Huang et al. (2021) (equivalently, Theorem 3 of Song et al. (2021)) to the joint space, and then apply additional variational inference to $\mathbf{v}_0$ with an application of Jensen's and change of measure. The second bound follows by analogy of the ISM-DSM connection from Huang et al. (2021); Song et al. (2021) for univariate diffusion models. □

We call the first bound the MDM ISM bound $\mathcal{L}^{\text{mism}}$ and the second the MDM DSM bound $\mathcal{L}^{\text{mdsm}}$. In practice, estimates of $\mathcal{L}^{\text{mism}}$ can be high variance and computationally prohibitive. We train MDM's by estimating the gradients of $\mathcal{L}^{\text{mdsm}}$. The ELBO requires sampling from the transition kernel $q_\phi(\mathbf{y}_s \mid x)$ and computing its score, an intractable problem for MDMs in general. Next, we provide two ingredients for tightening and optimizing these bounds in a generic fashion for linear inference MDMs.

3.3 INGREDIENT 1: COMPUTING THE TRANSITION $q_\phi(\mathbf{y}_s|x)$

To estimate eq. (6) and its gradients, we need samples from $q(\mathbf{y}_s|x)$ and compute $\nabla \log q(\mathbf{y}_s|x)$. We first show how to compute the mean and covariance of $q(\mathbf{y}_s|\mathbf{y}_0)$ and at the end of this section show how to adapt this to the mean and covariance of $q(\mathbf{y}_s|x)$. For linear MDMs, we can do so in closed form without requiring any diffusion-specific analysis. In this case, the transition kernel $q(\mathbf{y}_s|\mathbf{y}_0)$

is Gaussian (Särkkä & Solin, 2019). Let $f(\mathbf{y}, s) = \mathbf{A}(s)\mathbf{y}$. Then, the mean and covariance are solutions to the following ODEs:

$$d\mathbf{m}_{s|0}/ds = \mathbf{A}(s)\mathbf{m}_{s|0} \tag{7}$$

$$d\mathbf{\Sigma}_{s|0}/ds = \mathbf{A}(s)\mathbf{\Sigma}_{s|0} + \mathbf{\Sigma}_{s|0}\mathbf{A}^\top(s) + g^2(s). \tag{8}$$

The mean can be solved analytically:

$$\mathbf{m}_{s|0} = \exp\left[\int_0^s \mathbf{A}(\nu)d\nu\right]\mathbf{y}_0 \quad \underbrace{= \exp(s\mathbf{A})\mathbf{y}_0}_{\text{no integration if } \mathbf{A}(\nu) = \mathbf{A}} \quad . \tag{9}$$

The covariance equation does not have a standard form as eq. (9).

Instead of solving eq. (8) for a specific diffusion *manually*, as done in previous work (e.g. pages 50-54 of Dockhorn et al. (2021)), we show that a matrix factorization technique (Stengel, 1994; Särkkä & Solin, 2019) previously unused in diffusion-based generative models can automatically compute $\mathbf{\Sigma}_{s|0}$ generically for any linear MDM. We can define $\mathbf{C}_s, \mathbf{H}_s$ such that $\mathbf{\Sigma}_{s|0} = \mathbf{C}_s{\mathbf{H}_s}^{-1}$, where $\mathbf{C}_s, \mathbf{H}_s$ solve the following differential equation:

$$\begin{pmatrix} d\mathbf{C}_s/ds \\ d\mathbf{H}_s/ds \end{pmatrix} = \begin{pmatrix} \mathbf{A}(s) & g^2(s) \\ \mathbf{0} & -\mathbf{A}^\top(s) \end{pmatrix} \begin{pmatrix} \mathbf{C}_s \\ \mathbf{H}_s \end{pmatrix}. \tag{10}$$

These equations can be solved in closed-form,

$$\begin{pmatrix} \mathbf{C}_s \\ \mathbf{H}_s \end{pmatrix} = \exp\left[\begin{pmatrix} [\mathbf{A}]_s & [g^2]_s \\ \mathbf{0} & -[\mathbf{A}^\top]_s \end{pmatrix}\right] \begin{pmatrix} \mathbf{\Sigma}_0 \\ \mathbf{I} \end{pmatrix} \underbrace{= \exp\left[s\begin{pmatrix} \mathbf{A} & g^2 \\ \mathbf{0} & -\mathbf{A}^\top \end{pmatrix}\right] \begin{pmatrix} \mathbf{\Sigma_0} \\ \mathbf{I} \end{pmatrix}}_{\text{no integration if } \mathbf{A}(\nu) = \mathbf{A},\, g(\nu) = g}, \tag{11}$$

where $[\mathbf{A}]_s = \int_0^s \mathbf{A}(\nu)d\nu$. To condition on $\mathbf{y}_0 = (x, v)$, we set $\mathbf{\Sigma}_0 = \mathbf{0}$.

Computing $q_\phi(\mathbf{y}_s|x)$. For the covariance, to condition on x instead of $\mathbf{y}_0$, we set $\mathbf{\Sigma_0}$ to

$$\mathbf{\Sigma_0} = \begin{pmatrix} 0 & 0 \\ 0 & \mathbf{\Sigma_{v_0}} \end{pmatrix},$$

To compute the mean, it is the same expression as for $q(\mathbf{y}_s|\mathbf{y}_0)$, but with a different initial condition:

$$\mathbf{m}_{s|0} = \exp\left[\int_0^s \mathbf{A}(\nu)d\nu\right] \begin{pmatrix} x \\ \mathbb{E}_q[\mathbf{y}_0^v|x] \end{pmatrix} \tag{12}$$

See appendix C for more details.

A fast and simple algorithm. We show in algorithm 2 (appendix G) that computing the transition kernel only requires knowledge of f, g and requires no model-specific analysis. For $K-1$ auxiliary variables, $\mathbf{A}, g$ are $K \times K$ matrices. Similar to scalar diffusions, these parameters are shared across data coordinates. This means that the matrix exponentials and inverses are done on $K \times K$ matrices, where K is only 2 or 3 in our experiments. In table 1, we compare the time to sample a batch of size 256 from the transition kernel for CIFAR10 and MNIST. The table shows the extra computational cost of the automated algorithm is negligible.

Table 1: Runtime Comparison: we compare the run time of sampling from the CLD diffusion analytically versus using the automated algorithm.

Method	CIFAR-10	MNIST
Analytical	0.027	0.0062
Automated	0.029	0.007

3.4 Ingredient 2: MDM Parameterization

The MDM ELBO (eq. (6)) is tighter when the inference process distribution at time T tends toward the model's prior. Here we construct inference processes with the model prior π_θ as their stationary distribution q_∞.

Algorithm 1 Automatic Multivariate Diffusion Training

Input: Data $\{x_i\}$, inference process matrices $\mathbf{Q}_\phi, \mathbf{D}_\phi$, model prior π_θ, initial distribution $q_\phi(\mathbf{y}_0^v \mid x)$, and score model architecture s_θ
Returns: Trained score model s_θ
while s_θ not converged **do**
  Sample $x \sim \sum_{i=1}^N \frac{1}{N}\delta_{x_i}$, $v_0 \sim q_\phi(\mathbf{y}_0^v \mid x)$
  Sample $\mathbf{s} \sim \mathbf{U}[0, T]$ and $\mathbf{y}_s, \mathbf{y}_T \sim q_\phi(\mathbf{y}_s \mid x)$ using algorithm 2
  Estimate the stochastic gradient of the MDM ELBO, $\nabla_\theta \mathcal{L}(\theta, \phi)$, using eq. (6)
  $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}(\theta, \phi)$.
  if learning inference **then**
    $\phi \leftarrow \phi + \alpha \nabla_\phi \mathcal{L}(\theta, \phi)$
  end if
end while
Output s_θ

Ma et al. (2015) provide a complete recipe for constructing gradient-based MCMC samplers; they span all Markov diffusion processes with a given stationary distribution for the general non-linear case, which we extend to the time-varying processes. We specialize this to the linear case, where the parameterization simplifies to

$$d\mathbf{y} = \underbrace{-[\mathbf{Q}(s) + \mathbf{D}(s)]\nabla \mathbf{H}(\mathbf{y})}_{f}\, ds + \underbrace{\sqrt{2\mathbf{D}(s)}}_{g}\, d\widehat{\mathbf{B}}_s, \tag{13}$$

where $\mathbf{Q}$ is skew-symmetric ($-\mathbf{Q} = \mathbf{Q}^\top$), $\mathbf{D}$ is positive semi-definite and $\mathbf{H}(\mathbf{y}) = -\log q_\infty(\mathbf{y})$. Linearity means $\nabla \mathbf{H}$ must be linear and therefore q_∞ must be Gaussian. We show in appendix B.4 that existing diffusion processes such as VPSDE and CLD can also fit in the $\mathbf{Q}/\mathbf{D}$ parameterization. In appendix C, we show the transition kernel computation in the $\mathbf{Q}/\mathbf{D}$ parameterization, with an algorithm in Appendix G. We show the ELBO in terms of $\mathbf{Q}/\mathbf{D}$ in appendix F. Any skew-symmetric matrix $\mathbf{Q}$ can be written as $\mathbf{Q} = \mathbf{G} - \mathbf{G}^\top$ for an unconstrained parameter $\mathbf{G}$. We can set $\mathbf{D} = \mathbf{B}\mathbf{B}^\top$ for unconstrained parameter $\mathbf{B}$, which is always positive semi-definite. We provide more details on parameterization in appendix B.

Algorithm 1 shows the whole Automatic Multivariate Diffusion Training (AMDT) algorithm

3.5 Learning the inference process

The choice of diffusion matters, but the ELBOs in eq. (6) have no requirement for fixed inference. We therefore learn the inference process jointly with the model. Under linear transitions (ingredient 1), no algorithmic details change as the diffusion changes during learning. Under stationary parameterization (ingredient 2), we can learn without the stationary distribution going awry. Empirically, we observe that learning leads to improvements in likelihood lower-bounds, while also bypassing the need to specify a particular diffusion.

Looking at the MDM ELBO, $q_{\phi,\infty}$ may be set to equal π_θ, but it is $q_{\phi,T}$ for the chosen T that is featured in the ELBO. Learning the inference will reduce the cross-entropy:

$$-\mathbb{E}_{q_\phi(\mathbf{y}_T|x)}[\log \pi_\theta(\mathbf{y}_T)]. \tag{14}$$

Minimizing this term will tighten the ELBO for any score model. Next, q_ϕ is featured in the remaining DSM and ISM terms; optimizing for q_ϕ in those terms will tighten and improve the ELBO alongside the score model. Finally, q_ϕ is featured in the expectations and the $-\log q_\phi$ term:

$$\log p_\theta(\mathbf{u}_T^z = x) \geq \underbrace{\mathbb{E}_{q_\phi(\mathbf{y}_0^v = v|x)}}\left[(\text{ISM or DSM}) \underbrace{-\log q_\phi(\mathbf{y}_0^v = v)}\right] \tag{15}$$

These terms impose an optimality condition that $p_\theta(\mathbf{u}_T^v|\mathbf{u}_T^z) = q_\phi(\mathbf{y}_0^v|\mathbf{y}_0^z)$ (appendix D), When it is satisfied, no looseness in the ELBO for *the marginal likelihood of just the data* x is due to the initial time zero auxiliary variables.

The only change is that $\mathbf{Q}, \mathbf{D}$ need to be specified with parameters that enable gradients, and we have to use re-parameterized samples $\mathbf{y}_s|\mathbf{y}_0$.

4 INSIGHTS INTO MULTIVARIATE DIFFUSIONS

Scalar versus Multivariate Processes. The parameterization in eq. (13) clarifies what can change and what cannot to preserve the stationary distribution. Recall that $\mathbf{Q}$ and $\mathbf{D}$ are each $K \times K$ where there are $K-1$ auxiliary variables. Because 0 is the only 1×1 skew-symmetric matrix, this means that scalar processes must set $\mathbf{Q} = 0$. With a standard normal stationary distribution, the resulting process is:

$$d\mathbf{y} = -\mathbf{D}(s)\mathbf{y}ds + \sqrt{2\mathbf{D}(s)}d\widehat{\mathbf{B}}_s. \tag{16}$$

What is left is precisely the VPSDE process used widely in diffusion models where $\mathbf{D}(s) = \frac{1}{2}\beta(s)$ (Song et al., 2020b). This reveals that the VPSDE process is the only[3] scalar diffusion with a stationary distribution. This also clarifies the role of $\mathbf{Q}$: it accounts for mixing between dimensions in multivariate processes, as do non-diagonal entries in $\mathbf{D}$.

CLD optimizes a lower bound on the data log-likelihood. Looking at $\mathcal{L}^{\text{mdsm}}$ in eq. (6), taking the derivative with respect to the score model parameters, we show that the CLD objective formally maximizes a lower bound on the log marginal likelihood of the data x.

Does my model use auxiliary variables? In section 3 we gave the example choice of $q(\mathbf{y}_0^v|x) = q(\mathbf{y}_0^v) = \mathcal{N}(0,1)$ coordinate-wise. It is also common to set π_θ as independent standard Gaussians for all variables. Because the optimum for diffusions is $p_\theta = q$ for all t, the model has main and auxiliary dimensions independent at both endpoints 0 and T. Does this mean that the model does not use auxiliary variables? In appendix A, we show that even in this case, the model can use auxiliary variables at intermediate times. A sufficient condition is $\mathbf{Q} + \mathbf{D}$ is non-diagonal. On the other hand, when $\mathbf{Q} = \mathbf{0}$ and $\mathbf{D}$ is diagonal, under the independent endpoints, the model will not use the auxiliary variables to model the primary coordinate $\mathbf{u}_T^z$.

5 EXPERIMENTS

We test the MDM framework by testing two handcrafted diffusions and learned diffusions. The two handcrafted diffusions are (a) ALDA, introduced in Mou et al. (2019) as a scheme for accelerated gradient-based MCMC sampling (eq. (23)) and (b) MALDA: a modified version of ALDA (eq. (24)). Both of the specific diffusions have two auxiliary variables, see appendix B.4 for details. We also learn two diffusions with 1 and 2 auxiliary variables. We compare the performance of these four diffusions against VPSDE and ELBO-trained CLD.

Model	K	CIFAR-10	IMAGENET32	MNIST
CLD	2	≤ 3.106	3.566	≤ 1.34
VPSDE	1	≤ 3.461	3.710	≤ 1.418
VPSDE*	1	$\leq 3.08^*$	-	-
Learned	2	≤ 3.093	3.121	≤ 1.31
Learned	3	≤ 3.121	-	≤ 1.53
ALDA	3	≤ 7.99	-	≤ 10.882
MALDA	3	≤ 3.166	3.617	≤ 1.58

Table 2: BPD on CIFAR-10 and MNIST. $*$ indicates numbers from Song et al. (2021), which use a score model with **108 million** parameters. All other rows use a score model with **35.7 million** parameters for CIFAR-10 and 1.5 million for MNIST. The table shows that changes in the inference process can account for as much improvement as a 3x increase in score model parameters.

Following prior work, we train DBGMs on the CIFAR-10, IMAGENET32 and MNIST datasets for image generation. For both datasets, we use the modified U-Net architecture used in Ho et al. (2020); Huang et al. (2021). We input the auxiliary variables as extra channels, which leads to a negligible increase in parameter count. We use simple uniform dequantization; we do not tighten the

[3]There are processes such as sub-VPSDE (Song et al., 2020b) which are covered in the sense that they tend to members of this parameterization as T is large enough: sub-VP converges to VPSDE.

bounds with variational dequantization (Ho et al., 2019) or IWAE (Burda et al., 2015) for evaluation. We do not use exponential moving averages (Song & Ermon, 2020) for model parameters during evaluation. The evaluation in tables 2 and 3 is computed using $\mathcal{L}^{\text{mdsm}}$ (which reduces to the standard DSM ELBO for $K = 1$), where the integral over time is estimated by time points from the importance sampling distribution used in Song et al. (2021) and with truncation set to $\epsilon = 10^{-3}$. We report the ELBO using 1000 Monte Carlo samples of time per datapoint in the validation dataset.

MDMs versus VPSDE. Table 2 shows that the inference process matters. Despite smaller parameter counts using the score architecture from Ho et al. (2020), the MDM inference processes are able to achieve similar BPD compared to diffusions models with a significantly larger (3x increased) parameter count (the "baseline architecture" from Song et al. (2021)).

In table 2, we show that MDMs are able to achieve close to start of the art BPD for CIFAR10, with a significantly smaller number of parameters. In fig. 1 we plot the generated samples from CIFAR10 from the different DBGMs we have introduced.

Two-dimensional versus three-dimensional. Formally, the three-dimensional learned MDM includes fixed three-dimensional like MALDA, and learned two-dimensional MDMs as special cases. Learned two-dimensional MDMs include fixed two-dimensional diffusions like CLD and one-dimensional VPSDE processes as special cases. Yet we observe that learned two-dimensioanl MDMs peform better than learned or fixed three-dimensioanl MDMs. This is evidence that additional optimization considerations may improve allow three-dimensional MDMs beyond two-dimensional MDMs.

Figure 1: CIFAR10 samples generated from the learned and MALDA generative models.

6 Related Work

We now discuss several additional connections not already emphasized (as CLD is in the main text). Song et al. (2021); Huang et al. (2021) derive the ISM and DSM lower-bounds on the model log likelihood. Our work extends their analysis to the multivariate diffusion setting to derive lower bounds on the log marginal of the data in the presence of auxiliary variables.

Other work in generative modeling motivates auxiliary variables. Dupont et al. (2019) shows that operating on an augmented state space in neural ODEs allows a richer set of functions to modeled. Huang et al. (2020) use this principle for learning normalizing flows, where the authors maximize a joint likelihood $p_\theta(\mathbf{x}, \mathbf{e})$ in an extended space where $\mathbf{x} \sim q_{\text{data}}$ and $\mathbf{e} \sim \mathcal{N}(0, I_d)$ to model richer distributions over $\mathbf{x}$. Hierarchical variational inference and hierarchical variational auto-encoders marginalize variables auxiliary to build expressive inference distributions (Ranganath et al., 2016; Sønderby et al., 2016; Roberts et al., 2017; Hsu et al., 2018; Maaløe et al., 2019; Klushyn et al., 2019;

Model	CIFAR-10
FFJORD (Grathwohl et al., 2018)	3.40
Flow++ (Ho et al., 2019)	3.08
Gated PixelCNN (Van den Oord et al., 2016)	3.03
Sparse Transformer (Child et al., 2019)	2.83
VPSDE (Song et al., 2021)	≤ 3.08, $3.461^{\dagger}$
CLD (probability flow) (Dockhorn et al., 2021)	≤ 3.31
CLD (Dockhorn et al., 2021)	$\leq 3.106^{\dagger}$
Learned (2 vars)	≤ 3.093
Learned (3 vars)	≤ 3.121
ALDA (3 vars)	≤ 7.99
MALDA (3 vars)	≤ 3.166

Table 3: BPD on CIFAR-10. The first section displays existing models and existing evaluations. The † indicates DBGMs from existing work that we have trained. The second section displays existing models evaluated with the MDM ELBO that we derive in this work. The third section shows results for the new MDMs in this work as displayed in table 2. Unlike CLD evaluated with probability flow, MDMs trained and evaluated with their ELBO close the gap to existing flow and autoregressive results.

Vahdat & Kautz, 2020; Child, 2020). The work we present on diffusions with auxiliary variables can be seen as an application of this principle to DBGMs.

Learning the inference alongside the model is well-motivated in previous literature (Kingma & Welling, 2013) as well as in univariate diffusions (Sohl-Dickstein et al., 2015; Kingma et al., 2021). Kingma et al. (2021) show how to learn the inference process for VPSDE, that is learning the $\beta(s)$ function. For MDMs, $\mathbf{Q}$ can be non-zero; this shows that there are more choices to be made and learned in multivariate diffusions, compared to the scalar case. Additionally, one can choose the matrix $\mathbf{D}$ to be diagonal or full, give $\mathbf{Q}$ and $\mathbf{D}$ different time-varying functions, and learn $\nabla\mathbf{H}$.

Vahdat et al. (2021); Rombach et al. (2022) transform the data using auto-encoders and diffuse in a latent space. Future work is to use MDMs in concert with such encodings to yield effectively non-linear multivariate diffusions with automatic transitions kernels and tractable multivariate ELBOs.

7 Discussion

In this work, we present an algorithm for training multivariate diffusions for linear inference processes with a specified stationary distribution and any number of auxiliary variables, without any manual derivations required for training. This includes automating transition kernel computation and providing a parameterization of diffusions that have a specified stationary distribution. We also study learning the multivariate inference process. The experiments show that MDMs achieve BPDs similar to univariate diffusions, with as many as 3 times more score model parameters. The experiments also show that for the score models and datasets we study, MDMs outperform univariate diffusions. This work enables many directions to explore, including interactions across data and auxiliary coordinates, and improving optimization for diffusions with more auxiliary variables.

References

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, 2020.

Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images. *arXiv preprint arXiv:2011.10650*, 2020.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021.

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped langevin diffusion. *arXiv preprint arXiv:2112.07068*, 2021.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. *Advances in Neural Information Processing Systems*, 32, 2019.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pp. 2722–2730. PMLR, 2019.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23: 47–1, 2022.

Wei-Ning Hsu, Yu Zhang, Ron J Weiss, Heiga Zen, Yonghui Wu, Yuxuan Wang, Yuan Cao, Ye Jia, Zhifeng Chen, Jonathan Shen, et al. Hierarchical generative modeling for controllable speech synthesis. *arXiv preprint arXiv:1810.07217*, 2018.

Chin-Wei Huang, Laurent Dinh, and Aaron Courville. Augmented normalizing flows: Bridging the gap between generative flows and latent variable models. *arXiv preprint arXiv:2002.07101*, 2020.

Chin-Wei Huang, Jae Hyun Lim, and Aaron C Courville. A variational perspective on diffusion-based generative models and score matching. *Advances in Neural Information Processing Systems*, 34, 2021.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *arXiv preprint arXiv:2107.00630*, 2021.

Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke, and Patrick van der Smagt. Learning hierarchical priors in vaes. *Advances in neural information processing systems*, 32, 2019.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

Yi-An Ma, Tianqi Chen, and Emily Fox. A complete recipe for stochastic gradient mcmc. *Advances in neural information processing systems*, 28, 2015.

Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. Biva: A very deep hierarchy of latent variables for generative modeling. *Advances in neural information processing systems*, 32, 2019.

Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. *arXiv preprint arXiv:2103.16091*, 2021.

Wenlong Mou, Yi-An Ma, Martin J Wainwright, Peter L Bartlett, and Michael I Jordan. High-order langevin diffusion yields an accelerated mcmc algorithm. *arXiv preprint arXiv:1908.10859*, 2019.

Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Rajesh Ranganath, Dustin Tran, and David Blei. Hierarchical variational models. In *International conference on machine learning*, pp. 324–333. PMLR, 2016.

Adam Roberts, Jesse Engel, and Douglas Eck. Hierarchical variational autoencoders for music. In *NIPS Workshop on Machine Learning for Creativity and Design*, volume 3, 2017.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

Simo Särkkä and Arno Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

Hiroshi Sasaki, Chris G Willcocks, and Toby P Breckon. Unit-ddpm: Unpaired image translation with denoising diffusion probabilistic models. *arXiv preprint arXiv:2104.05358*, 2021.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. *Advances in neural information processing systems*, 29, 2016.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *arXiv preprint arXiv:2006.09011*, 2020.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pp. 574–584. PMLR, 2020a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34:1415–1428, 2021.

Robert F Stengel. *Optimal control and estimation*. Courier Corporation, 1994.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 33:19667–19679, 2020.

Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *Advances in Neural Information Processing Systems*, 34:11287–11302, 2021.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

A DOES MY MODEL USE AUXILIARY VARIABLES?

In section 3 we gave the example choice of $q(\mathbf{y}_0^v|x) = q(\mathbf{y}_0^v) = \mathcal{N}(0,1)$ coordinate-wise. It is also a common choice to set π_θ as independent standard Gaussians for all dimensions. Because the optimum in diffusion models is $p_\theta = q$ for all t, we see a peculiar phenomenon under this choice: the model has main and auxiliary dimensions independent at both endpoints 0 and T. Does this mean that the model does not use auxiliary variables? We show that even when $q_\phi(\mathbf{y}_0)$ and π_θ have main and auxiliary variables independent, the model can use the auxiliary variables. A sufficient condition is $\mathbf{Q} + \mathbf{D}$ is non-diagonal.

To make this precise, we recall that we model with $p_\theta(\mathbf{u}_T^z = x)$. To show the model is using auxiliary variables, we just need to show that $\mathbf{u}_T^z$ (main coordinate at T) depends on $\mathbf{u}_t^v$ (aux. coordinate at t) for $T > t$. At optimum, $p_\theta(\mathbf{u}_T^z, \mathbf{u}_t^v) = q_\phi(\mathbf{y}_0^z, \mathbf{y}_{T-t}^v)$. Therefore it is sufficient to show that for some time s, $q_\phi(\mathbf{y}_s^v|\mathbf{y}_0^z) \neq q_\phi(\mathbf{y}_s^v)$. Because $\mathbf{y}_0^z$, is determined by x we need to show that $q_\phi(\mathbf{y}_s^v|x) \neq q_\phi(\mathbf{y}_s^v)$. To do that, we first derive $q(\mathbf{y}_s|x)$ and then marginalize to get $q(\mathbf{y}_s^v|x)$ from it. Since the former is 2D gaussian, the latter is available in terms of the former's mean and covariance. Suppose $\mathbb{E}[\mathbf{y}_0^v] = 0$, $\mathbf{Q} = [[0,-1],[1,0]]$ and $\mathbf{D} = [[1,0],[0,1]]$ and we have $s = .1$ We have:

$$\mathbb{E}[\mathbf{y}_s|x] = \exp\Big[-s(\mathbf{Q}+\mathbf{D})\Big]\begin{pmatrix} x \\ 0 \end{pmatrix} = \exp\Big[\begin{bmatrix} -.1 & .1 \\ -.1 & -.1 \end{bmatrix}\Big]\begin{pmatrix} x \\ 0 \end{pmatrix} = \begin{pmatrix} 0.9003x \\ -0.090x \end{pmatrix} \tag{17}$$

Regardless of the covariance any 1D of this 2D gaussian will have mean that is a function of x, meaning that $q(\mathbf{y}_s^v|x)$ does not equal $q(\mathbf{y}_s^v)$ (which is also a Gaussian but with mean depending on $x's$ mean rather than x itself. Therefore, even under the setup with independent endpoints, the optimal model makes use of the intermediate auxiliary variables in its final modeling distribution $p_\theta(\mathbf{u}_T^z = x)$.

Are there choices of $\mathbf{Q}$ and $\mathbf{D}$ that lead to learning models that don't make use of the extra dimensions? As mentioned, in the inference process, $\mathbf{Q}$ is responsible for mixing information among the coordinates, and is the only source of this when $\mathbf{D}$ is diagonal. Then, if $\mathbf{Q} = \mathbf{0}$ and $\mathbf{D}$ is diagonal, none of the coordinates for a given feature $\mathbf{x}_j$ (including $\mathbf{u}_{tj}^z, \mathbf{u}_{tj}^{v_1}, \ldots, \mathbf{u}_{tj}^{v_{K-1}}$) interact for any t. Then, since $p_\theta = q$ at optimum, independence of the coodinates at all t in q imply the same in p_θ and the model will not make use of any auxiliary variables when modeling the marginal $\log p_\theta(\mathbf{u}_T^z = x)$.

B STATIONARY PARAMETERIZATION

The general process family

$$d\mathbf{y} = f(\mathbf{y}, s)ds + g(s)d\widehat{\mathbf{B}}_s \tag{18}$$

can be restricted to those with stationary distributions. Ma et al. (2015) show a complete recipe to span all non-linear processes that have a given stationary distribution. We first observe that their formulation also applies to time-varying processes, though this is not mentioned in that work. We then restrict this to linear processes (per the other requirement for tractable transition kernels). The parameterization from Ma et al. (2015) specialized to linear processes is

$$d\mathbf{y} = \underbrace{-[\mathbf{Q}(s) + \mathbf{D}(s)]\nabla\mathbf{H}(\mathbf{y})}_{f}\,ds + \underbrace{\sqrt{2\mathbf{D}(s)}}_{g}\,d\widehat{\mathbf{B}}_s \tag{19}$$

where $\mathbf{Q}$ is skew-symmetric ($-\mathbf{Q} = \mathbf{Q}^\top$), $\mathbf{D}$ is positive semi-definite and $\mathbf{H}(\mathbf{y})$ determines the stationary distribution with $\mathbf{H}(\mathbf{y}) = -\log q_\infty(\mathbf{y})$. For a linear process $\nabla\mathbf{H}$ has to be linear, meanining these processes are all multivariate normal. This parameterization spans all linear Markov processes that have stationary distributions.

B.1 PARAMETERIZING $\mathbf{Q}_\phi$

Suppose $b_q(s)$ is a positive scalar function defined on the time domain with known integral. Suppose $\tilde{\mathbf{Q}}_\phi$ is any matrix. Then $\tilde{\mathbf{Q}}_\phi - \tilde{\mathbf{Q}}_\phi^\top$ is skew-symmetric with $\tilde{\mathbf{Q}}_{\phi,ij} = -\tilde{\mathbf{Q}}_{\phi,ji}$. We can set $\mathbf{Q}_\phi$ to

$$\mathbf{Q}_\phi(s) = b_q(s) \cdot \Big[\tilde{\mathbf{Q}}_\phi - \tilde{\mathbf{Q}}_\phi^\top\Big] \tag{20}$$

This is a general parameterization of time-independent skew-symmetric matrices, which have number of degrees of freedom equal to the number of entries in one of the triangles of the matrix, excluding the diagonal.

B.2 PARAMETERIZING $\mathbf{D}_\phi$

Suppose $b_d(s)$ is a positive scalar function defined on the time domain with known integral. Suppose $\tilde{\mathbf{D}}_\phi$ is any matrix. Then $\tilde{\mathbf{D}}_\phi \tilde{\mathbf{D}}_\phi^\top$ is positive semi-definite and spans all time-independent skew-symmetric matrices. We can set $\mathbf{D}_\phi$ to

$$\mathbf{D}_\phi(s) = b_d(s) \cdot \left[\tilde{\mathbf{D}}_\phi \tilde{\mathbf{D}}_\phi^\top\right] \tag{21}$$

To show $\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top$ spans all positive semi-definite matrices: suppose $\mathbf{M}$ is positive semi-definite. Then it is square. Then it can be eigen-decomposed into $\mathbf{M} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^\top$ The degrees of freedom in $\mathbf{V}\mathbf{\Sigma}\mathbf{V}^\top$ are just $\mathbf{R} = \mathbf{V}\sqrt{\mathbf{\Sigma}}$ since $\mathbf{V}\mathbf{\Sigma}\mathbf{V}^\top = \mathbf{R}\mathbf{R}^\top$ and the square root is taken element-wise because $\mathbf{\Sigma}$ is diagonal and is real because each $\mathbf{\Sigma}_{ij} \geq 0$, which is true because $\mathbf{M}$ is positive semi-definite. Take $\mathbf{D} = \mathbf{R}$.

A simpler option is to just parameterize a non-negative vector to use as the diagonal of a diagonal-only matrix. But we stick with the fuller approach.

B.3 INTEGRALS

The known integral requirement comes from the integrals required in the transition kernel, and can be relaxed two possible ways:

- numerical integration of function with unknown integral, which might be okay since the functions are scalar-in scalar-out
- Directly parameterize the integral and use auto-grad when needing the functions not-integrated.

We stick with the known integrals. In conclusion, the underlying parameters are positive scalar functions $b_q(s), b_d(s)$ defined on the time domain and with known integral, and general matrices $\tilde{\mathbf{Q}}_\phi, \tilde{\mathbf{D}}_\phi$.

B.4 INSTANCES

CLD. The CLD process (eq 5 in Dockhorn et al. (2021)) is defined as

$$\begin{pmatrix} d\mathbf{x}_t \\ d\mathbf{v}_r \end{pmatrix} = d\mathbf{u}_t = \begin{pmatrix} 0 & \frac{\beta}{M} \\ -\beta & -\frac{\Gamma\beta}{M} \end{pmatrix} \mathbf{u}_t + \begin{pmatrix} 0 & 0 \\ 0 & \sqrt{2\Gamma\beta} \end{pmatrix} d\tilde{\mathbf{w}}_t.$$

In Q, D, we have

$$H(\mathbf{u}) = \frac{1}{2}\|\mathbf{x}\|_2^2 + \frac{1}{2M}\|\mathbf{v}\|_2^2, \qquad \nabla_{\mathbf{u}} H(\mathbf{u}) = \begin{pmatrix} \mathbf{x} \\ \frac{1}{M}\mathbf{v} \end{pmatrix}$$
$$Q = \begin{pmatrix} 0 & -\beta \\ \beta & 0 \end{pmatrix}, \qquad D = \begin{pmatrix} 0 & 0 \\ 0 & \Gamma\beta \end{pmatrix}$$

And the stationary distribution is

$$q_{\phi,\infty} \propto \exp(-H(\mathbf{u})) = \mathcal{N}(\mathbf{x}; 0, I_d)\mathcal{N}(\mathbf{v}; 0, MI_d) \tag{22}$$

ALDA. Mou et al. (2019) define a third-order diffusion process for the purpose of gradient-based MCMC sampling. The ALDA diffusion process can be specified as

$$Q = \begin{pmatrix} 0 & -\frac{1}{L}I & 0 \\ \frac{1}{L}I & 0 & -\gamma I \\ 0 & \gamma I & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{\xi}{L}I \end{pmatrix}. \tag{23}$$

Note that Q is skew-symmetric and D is positive semi-definite, therefore we have that $p_t(\mathbf{u}) \to$. In this case this is

$$q_{\phi,\infty} = \mathcal{N}(0, I_d)\mathcal{N}(\mathbf{z}; 0, \frac{1}{L}I_d)\mathcal{N}(\mathbf{r}; 0, \frac{1}{L}I_d)$$

MALDA. Similar to ALDA, we specify a diffusion process we term MALDA which we specify as

$$Q = \begin{pmatrix} 0 & -\frac{1}{L}I & -\frac{1}{L} \\ \frac{1}{L}I & 0 & -\gamma I \\ \frac{1}{L} & \gamma I & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{L}I & 0 \\ 0 & 0 & \frac{1}{L}I \end{pmatrix}. \tag{24}$$

Note that Q is skew-symmetric and D is positive semi-definite. In this case this is

$$q_{\phi,\infty} = \mathcal{N}(0, I_d)\mathcal{N}(\mathbf{v}_1; 0, \frac{1}{L}I_d)\mathcal{N}(\mathbf{v}_2; 0, \frac{1}{L}I_d)$$

C TRANSITIONS FOR LINEAR PROCESSES

For time variable s and Brownian motion $\widehat{\mathbf{B}}_s$ driving diffusions of the form

$$d\mathbf{y} = f(\mathbf{y}, s)ds + g(s)d\widehat{\mathbf{B}}_s, \tag{25}$$

when $f_\phi(\mathbf{y}_s, s), g_\phi(s)$ are linear, we can sample/compute transitions in closed form. The transition kernel $q_\phi(\mathbf{y}_s|\mathbf{y}_0)$ is always normal (Särkkä & Solin, 2019). Therefore, we just find the mean $\mathbf{m}_{s|0}$ and covariance $\mathbf{\Sigma}_{s|0}$ of $q(\mathbf{y}_s|\mathbf{y}_0)$. Let $f(\mathbf{y}, s) = \mathbf{A}(s)\mathbf{y}$. The un-conditional time s mean and covariance are solutions to

$$d\mathbf{m}_s/ds = \mathbf{A}(s)\mathbf{m}_s \tag{26}$$

$$d\mathbf{\Sigma}_s/ds = \mathbf{A}(s)\mathbf{\Sigma}_s + \mathbf{\Sigma}_s\mathbf{A}^\top(s) + g^2(s) \tag{27}$$

By (6.6) in the book for computing conditionals $q(\mathbf{y}_s|\mathbf{y}_0)$, we can take the marginal distribution ODEs and compute conditionals by simply setting the time 0 mean and covariance initial conditions to the conditioning value and to $\mathbf{0}$ respectively. We take (6.36-6.39) and set $\mathbf{m}_0 = \mathbf{u}_0$ and $\mathbf{\Sigma}_0 = 0$ to condition. Let $[\mathbf{A}]_s = \int_0^s \mathbf{A}(\nu)d\nu$. The mean is

$$\mathbf{m}_{s|0} = \exp\left[\int_0^s \mathbf{A}(\nu)d\nu\right]\mathbf{y}_0 = \exp\left(\Big[A\Big]_s\right) \underbrace{= \exp(s\mathbf{A})\mathbf{y}_0}_{\text{no integration if } \mathbf{A}(\nu) = \mathbf{A}}, \tag{28}$$

where exp denotes matrix exponential. (6.36-6.39) show how to express the covariance $q(\mathbf{y}_s|\mathbf{y}_0)$ as a matrix factorization[4] $\mathbf{\Sigma}_s = \mathbf{C}_s(\mathbf{H}_s)^{-1}$ for $\mathbf{C}_s, \mathbf{H}_s$ being the solutions of:

$$\begin{pmatrix} \frac{d}{ds}\mathbf{C}_s \\ \frac{d}{ds}\mathbf{H}_s \end{pmatrix} = \begin{pmatrix} \mathbf{A}(s) & g^2(s) \\ \mathbf{0} & -\mathbf{A}^\top(s) \end{pmatrix}\begin{pmatrix} \mathbf{C}_s \\ \mathbf{H}_s \end{pmatrix} \tag{29}$$

To condition and get $\mathbf{\Sigma}_{s|0}$ from $\mathbf{\Sigma}_s$, we set $\mathbf{\Sigma}_0 = \mathbf{0}$. To satisfy $\Sigma_0 = \mathbf{C}_0({\mathbf{H}_0}^{-1}) = \mathbf{0}$ we set $\mathbf{C}_0 = \mathbf{0}$ and $\mathbf{H}_0 = \mathbf{I}$.

$$\begin{pmatrix} \mathbf{C}_s \\ \mathbf{H}_s \end{pmatrix} = \exp\left[\begin{pmatrix} [\mathbf{A}]_s & [g^2]_s \\ \mathbf{0} & -[\mathbf{A}^\top]_s \end{pmatrix}\right]\begin{pmatrix} \mathbf{0} \\ \mathbf{I} \end{pmatrix} \underbrace{= \exp\left[s\begin{pmatrix} \mathbf{A} & g^2 \\ \mathbf{0} & -\mathbf{A}^\top \end{pmatrix}\right]\begin{pmatrix} \mathbf{0} \\ \mathbf{I} \end{pmatrix}}_{\text{no integration if } \mathbf{A}(\nu) = \mathbf{A},\, g(\nu) = g}. \tag{30}$$

Finally, $\mathbf{\Sigma}_{s|0} = \mathbf{C}_s(\mathbf{H}_s)^{-1}$.

C.1 HYBRID SCORE MATCHING

Instead of computing $q(\mathbf{y}_s|\mathbf{y}_0)$, we can apply the hybrid score matching principle (Dockhorn et al., 2021) to reduce variance by compute objectives using $q(\mathbf{y}_s|x)$ instead of $q(\mathbf{y}_s|\mathbf{y}_0)$, which amounts

[4] We rename their $\mathbf{C}, \mathbf{D}$ to $\mathbf{C}, \mathbf{H}$ to avoid mixing up with $\mathbf{D}$ in our $\mathbf{Q}, \mathbf{D}$ stationary parameterization.

to integrating out $\mathbf{v}_0$. Formally, one can show the equality of the ELBO with $q(\mathbf{y}_s|\mathbf{y}_0)$ swapped-out for $q(\mathbf{y}_s|x)$. To accomplish this, following Särkkä & Solin (2019), we simply replace $\mathbf{y}_0$ with $[x, \mathbb{E}[\mathbf{v}_0]]$ in the expression for $\mathbf{m}_{s|0}$, i.e. replace the conditioning value of $\mathbf{v}_0$ with the mean of its chosen initial distribution:

$$\mathbb{E}[\mathbf{y}_s|x] = \exp\left[\int_0^s A(\nu)d\nu\right]\begin{pmatrix} x \\ \mathbb{E}[\mathbf{v}_0]\end{pmatrix} \tag{31}$$

For the convariance, instead of using $\mathbf{C}_0 = \mathbf{\Sigma}_0 = \mathbf{0}$, we use a block matrix to condition on x but not $\mathbf{v}_0$. We decompose $\mathbf{\Sigma}_0$ into its blocks $\mathbf{\Sigma}_{0,xx}$, $\mathbf{\Sigma}_{0,vv}$,$\mathbf{\Sigma}_{0,xv}$. As before, to condition on x we set $\mathbf{\Sigma}_{0,xx} = \mathbf{0}$. Because $q(\mathbf{v}_0)$ is set to be independent of x, $\mathbf{\Sigma}_{0,xv}$ is also set to $\mathbf{0}$. Finally, instead of $\mathbf{0}$, to marginalize out $\mathbf{v}_0$, $\mathbf{\Sigma}_{0,vv}$ is set to the covariance of the chosen initial time zero distribution for $\mathbf{v}_0$. E.g. if $\mathbf{v}_{0,j} \sim N(0, \gamma)$ for each dimension, then $\mathbf{\Sigma}_{0,vv} = N(0, \gamma I)$.

We operationalize this in a simple piece of code, which makes the ELBO tractable and easy, i.e. skips both analytic derivations and numerical forward integration during training.

C.2 Transitions in Stationary Parameterization

In terms of $\mathbf{Q}, \mathbf{D}$, the transitions $q(\mathbf{y}_s|\mathbf{y}_0)$ for time s are normal with mean $\mathbf{m}_{s|0}$ and $\mathbf{\Sigma}_{s|0}$ equal to:

$$\mathbf{m}_{s|0} = \exp\Big(-\Big[\mathbf{Q}+\mathbf{D}\Big]_s\Big)\mathbf{y}_0, \qquad \begin{pmatrix}\mathbf{C}_s \\ \mathbf{H}_s\end{pmatrix} = \exp\left[\begin{pmatrix} -[\mathbf{Q}+\mathbf{D}]_s & [2\mathbf{D}]_s \\ \mathbf{0} & [(\mathbf{Q}+\mathbf{D})^\top]_s\end{pmatrix}\right]\begin{pmatrix}\mathbf{0} \\ \mathbf{I}\end{pmatrix} \tag{32}$$

where $\mathbf{\Sigma}_{s|0} = \mathbf{C}_s(\mathbf{H}_s)^{-1}$. For the time invariant case, this simplifies to

$$\mathbf{m}_{s|0} = \exp[-s(\mathbf{Q}+\mathbf{D})]\mathbf{y}_0, \qquad \begin{pmatrix}\mathbf{C}_s \\ \mathbf{H}_s\end{pmatrix} = \exp\left[s\begin{pmatrix} -(\mathbf{Q}+\mathbf{D}) & 2\mathbf{D} \\ \mathbf{0} & (\mathbf{Q}+\mathbf{D})^\top\end{pmatrix}\right]\begin{pmatrix}\mathbf{0} \\ \mathbf{I}\end{pmatrix} \tag{33}$$

D Generic change of measure and Jensen's for approximate marginalization

Suppose $\mathbf{u} = [\mathbf{z}, \mathbf{v}]$ and we have an expression for $p(\mathbf{u} = [z, v]) = p(\mathbf{z} = z, \mathbf{v} = v)$. By marginalization, we can get $p(\mathbf{z} = z)$, and we can introduce another distribution q to pick a sampling distribution of our choice:

$$\begin{aligned} p(\mathbf{z} = z) &= \int_v p(\mathbf{z} = z, \mathbf{v} = v)dv \\ &= \int_v p(\mathbf{z} = z|\mathbf{v} = v)p(\mathbf{v} = v)dv \\ &= \int_v \frac{q(\mathbf{v} = v|\mathbf{z} = z)}{q(\mathbf{v} = v|\mathbf{z} = z)}p(\mathbf{z} = z|\mathbf{v} = v)p(\mathbf{v} = v)dv \\ &= \mathbb{E}_{q(\mathbf{v}=v|\mathbf{z}=z)}\Big[\frac{p(\mathbf{z} = z, \mathbf{v} = v)}{q(\mathbf{v} = v|\mathbf{z} = z)}\Big] \end{aligned} \tag{34}$$

We often work with these expressions in log space, and need to pull the expectation outside to use monte carlo. Jensen's bound allows us to do this:

$$\begin{aligned} \log p(\mathbf{z} = z) &= \log \mathbb{E}_{q(\mathbf{v}=v|\mathbf{z}=z)}\Big[\frac{p(\mathbf{z} = z, \mathbf{v} = v)}{q(\mathbf{v} = v|\mathbf{z} = z)}\Big] \\ &\geq \mathbb{E}_{q(\mathbf{v}=v|\mathbf{z}=z)}\Big[\log\frac{p(\mathbf{z} = z, \mathbf{v} = v)}{q(\mathbf{v} = v|\mathbf{z} = z)}\Big] \end{aligned}$$

The following shows us that the bound is tight when $q(\mathbf{v} = v|\mathbf{z} = z) = p(\mathbf{v} = v|\mathbf{z} = z)$:

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{v}=v|\mathbf{z}=z)}\left[\log \frac{p(\mathbf{z}=z, \mathbf{v}=v)}{q(\mathbf{v}=v|\mathbf{z}=z)}\right] &=_{\text{assume}} \mathbb{E}_{p(\mathbf{v}=v|\mathbf{z}=z)}\left[\log \frac{p(\mathbf{z}=z, \mathbf{v}=v)}{p(\mathbf{v}=v|\mathbf{z}=z)}\right] \\
&= \mathbb{E}_{p(\mathbf{v}=v|\mathbf{z}=z)}\left[\log \left(\frac{p(\mathbf{z}=z, \mathbf{v}=v)}{p(\mathbf{v}=v, \mathbf{z}=z)} \cdot p(\mathbf{z}=z)\right)\right] \\
&= \mathbb{E}_{p(\mathbf{v}=v|\mathbf{z}=z)}\left[\log p(\mathbf{z}=z)\right] \\
&= \log p(\mathbf{z}=z)
\end{aligned} \tag{35}$$

This is similar to the behavior of q in the traditional ELBO (Blei et al., 2017).

E ELBO FOR MDMS

$$\log p_\theta(x) = \log \int_{v_0} p_\theta(x_0, v_0) dv_0 \tag{36}$$

$$= \log \int_{v_0} p_\theta(u_0 = [x, v_0]) \tag{37}$$

$$= \log \int_{v_0} \frac{q(v_0|x)}{q(v_0|x)} p_\theta(u_0 = [x, v_0]) \tag{38}$$

$$= \log \mathbb{E}_{q(v_0|x)}\left[\frac{p_\theta(u_0 = [x, v_0])}{q(v_0|x)}\right] \tag{39}$$

$$\geq \mathbb{E}_{q(v_0|x)}\left[\log p_\theta(u_0 = [x, v_0]) - \log q(v_0|x)\right] \tag{40}$$

$$\geq \mathbb{E}_{q(y|x)}\left[\log \pi_\theta(y_T) + \int_0^T -\|s_\theta\|_{g^2}^2 - \nabla \cdot (g^2 s_\theta - f) ds - \log q(y_0^v|x)\right] \tag{41}$$

The first inequality holds due to Jensen's inequality and the second due to an application of Theorem 1 from Huang et al. (2021) or Theorem 3 from Song et al. (2021) applied to the joint variable $\mathbf{u}_0$.

E.1 ISM TO DSM

E.1.1 LEMMA: EXPECTATION BY PARTS

We will need a form of multivariate integration by parts which gives us for some f and some $q(x)$, $E_{q(x)}[\nabla_x \cdot f(x)] = -E_{q(x)}[f(x)^\top \nabla_x \log q(x)]$

$$
\begin{aligned}
E_{q(x)}[\nabla_x \cdot f_i(x)] &= \int q(x) \sum_{i=1}^{d} [\nabla_{x_i} f_i(x)] dx \\
&= \int \sum_{i=1}^{d} q(x) \nabla_{x_i} f_i(x) dx \\
&= \sum_{i=1}^{d} \int_{x_{-i}} \int_{x_i} q(x) \nabla_{x_i} f_i(x) dx_i dx_{-i} \\
&= \sum_{i=1}^{d} \int \left[\Big[q(x) \int \nabla_{x_i} f_i(x) dx_i \Big]_{-\infty}^{\infty} - \int \nabla_{x_i} q(x) \int \nabla_{x_i} f_i(x) dx_i] \right] dx_{-i} \\
&= \sum_{i=1}^{d} \int \left[- \int \nabla_{x_i} q(x) f_i(x) dx_i \right] dx_{-i} \\
&= \sum_{i=1}^{d} \int \left[- \int q(x) \nabla_{x_i} \log q(x) f_i(x) dx_i \right] dx_{-i} \\
&= \sum_{i=1}^{d} - \int \int q(x) \nabla_{x_i} \log q(x) f_i(x) dx_i dx_{-i} \\
&= \sum_{i=1}^{d} -E_{q(x)} \Big[\nabla_{x_i} \log q(x) f_i(x) \Big] \\
&= -E_{q(x)} [f(x)^\top \nabla_x \log q(x)]
\end{aligned}
$$

E.1.2 DSM ELBO

Using the "expectation by parts", we have:

$$
\mathbb{E}_{q(u_t|x)}[\nabla_{u_t} \cdot g^2(t) s_\theta(u_t, t)] = -\mathbb{E}_{q(u_t|x)}[(g^2(t) s_\theta(u_t, t))^\top \nabla_{u_t} \log q(u_t|x)]
$$

Also we have, for s_θ evaluated at (u_t, t), by completing the square,

$$
-\frac{1}{2}||s_\theta||_{g^2(t)} + s_\theta^\top g^2(t) \nabla \log q(u_t|x) = -\frac{1}{2}||s_\theta - \nabla \log q(u_t|x)||^2_{g^2(t)} + .5||\nabla \log q(u_t|x)||^2_{g^2(t)}
$$

The two together give us:

$$
\begin{aligned}
\log p(x) &\geq \mathbb{E}_{q(u_T|x)} \left[\log \pi \right] + \int_0^T \left[\mathbb{E}_{q(u_t|x)} \Big[-\nabla \cdot g^2 s_\theta - .5||s_\theta||^2_{g^2(t)} + \nabla \cdot f \Big] dt \right] \\
&= \mathbb{E}_{q(u_T|x)} \left[\log \pi \right] + \int_0^T \left[\mathbb{E}_{q(u_t|x)} \Big[(g^2 s_\theta)^\top \nabla_{u_t} \log q(u_t|x) - .5||s_\theta||^2_{g^2(t)} + \nabla \cdot f \Big] dt \right] \\
&= \mathbb{E}_{q(u_T|x)} \left[\log \pi \right] + \int_0^T \left[\mathbb{E}_{q(u_t|x)} \Big[-\frac{1}{2}||s_\theta - \nabla \log q(u_t|x)||^2_{g^2(t)} + .5||\nabla \log q(u_t|x)||^2_{g^2(t)} + \nabla_{u_t} \cdot f \Big] \right] dt
\end{aligned}
\tag{42}
$$

E.2 NOISE PREDICTION

We have that for normal $\mathcal{N}(\mathbf{y}_s; \mathbf{m}_{s|0}, \mathbf{\Sigma}_{s|0})$, we can sample $\mathbf{y}_s$ with normal noise $\epsilon \sim \mathcal{N}(0, I)$ and $\mathbf{y}_s = \mathbf{m}_{s|0} + \mathbf{L}\epsilon$ where $\mathbf{L}$ is the cholesky decomposition of $\mathbf{\Sigma}_{s|0}$ Then, the score is

$$\nabla_{\mathbf{y}_s} \log q(\mathbf{y}_s|\mathbf{y}_0)\Big|_{\mathbf{y}_s=\mathbf{m}_{s|0}+\mathbf{L}\epsilon} \tag{43}$$

$$= -\mathbf{\Sigma}_{s|0}^{-1}\Big(\mathbf{y}_s - \mathbf{m}_{s|0}\Big) \tag{44}$$

$$= -\mathbf{\Sigma}_{s|0}^{-1}\Big(\Big[\mathbf{m}_{s|0} + \mathbf{L}\epsilon\Big] - \mathbf{m}_{s|0}\Big) \tag{45}$$

$$= -\mathbf{\Sigma}_{s|0}^{-1}\Big(\mathbf{L}\epsilon\Big) \tag{46}$$

$$= -\Big(\mathbf{L}\mathbf{L}^\top\Big)^{-1}\Big(\mathbf{L}\epsilon\Big) \tag{47}$$

$$= -\Big(\mathbf{L}^\top\Big)^{-1}\mathbf{L}^{-1}\mathbf{L}\epsilon \tag{48}$$

$$= -\Big(\mathbf{L}^\top\Big)^{-1}\epsilon = -\Big(\mathbf{L}^{-1}\Big)^\top\epsilon = -\mathbf{L}^{\top,-1}\epsilon \tag{49}$$

So lets parameterize $s_\theta(\mathbf{y}_s, s)$ as $s_\theta(\mathbf{y}_s, s) = -\mathbf{L}^{\top,-1}\epsilon_\theta(\mathbf{y}, s)$. This gives

$$\frac{1}{2}\| - \mathbf{L}^{\top,-1}\epsilon_\theta(\mathbf{y}, s) \quad - \quad -\mathbf{L}^{\top,-1}\epsilon\|^2_{g^2_\phi(s)} \tag{50}$$

$$= \frac{1}{2}\|\mathbf{L}^{\top,-1}\epsilon \quad - \quad \mathbf{L}^{\top,-1}\epsilon_\theta(\mathbf{y}, s)\|^2_{g^2_\phi(s)} \tag{51}$$

$$= \frac{1}{2}\Bigg(\mathbf{L}^{\top,-1}\epsilon \quad - \quad \mathbf{L}^{\top,-1}\epsilon_\theta(\mathbf{y}, s)\Bigg)^\top g^2_\phi(s)\Bigg(\mathbf{L}^{\top,-1}\epsilon \quad - \quad \mathbf{L}^{\top,-1}\epsilon_\theta(\mathbf{y}, s)\Bigg) \tag{52}$$

$$= \frac{1}{2}\Bigg(\mathbf{L}^{\top,-1}\Big[\epsilon - \epsilon_\theta(\mathbf{y}, s)\Big]\Bigg)^\top g^2_\phi(s)\Bigg(\mathbf{L}^{\top,-1}\Big[\epsilon - \epsilon_\theta(\mathbf{y}, s)\Big]\Bigg) \tag{53}$$

We can also use this insight to lower the variance of the dsm's quadratic term:

$$\mathbb{E}_{\mathbf{y}_0}\mathbb{E}_{\mathbf{y}_s|\mathbf{y}_0}\Bigg[\frac{1}{2}\|\nabla_{\mathbf{y}_s}\log q_\phi(\mathbf{y}_s|\mathbf{y}_0)\|^2_{g^2_\phi(s)}\Bigg] = \mathbb{E}_{\mathbf{y}_0}\mathbb{E}_{\mathbf{y}_s|\mathbf{y}_0}\Bigg[\Big(\nabla_{\mathbf{y}_s}\log q_\phi(\mathbf{y}_s|\mathbf{y}_0)\Big)^\top g^2_\phi(s)\Big(\nabla_{\mathbf{y}_s}\log q_\phi(\mathbf{y}_s|\mathbf{y}_0)\Big)\Bigg] \tag{54}$$

$$= \mathbb{E}_{\mathbf{y}_0}\mathbb{E}_{\mathbf{y}_s|\mathbf{y}_0}\Bigg[\Big(-\mathbf{L}^{\top,-1}\epsilon\Big)^\top g^2_\phi(s)\Big(-\mathbf{L}^{\top,-1}\epsilon\Big)\Bigg] \tag{55}$$

$$= \mathbb{E}_{\mathbf{y}_0}\mathbb{E}_{\mathbf{y}_s|\mathbf{y}_0}\Bigg[\epsilon^\top(-\mathbf{L}^{-1})g^2_\phi(s)(-\mathbf{L}^{\top,-1})\epsilon\Bigg] \tag{56}$$

$$= \mathbb{E}_{\mathbf{y}_0}\mathbb{E}_{\mathbf{y}_s|\mathbf{y}_0}\Bigg[\epsilon^\top\Big(\mathbf{L}^{-1}g^2_\phi(s)\mathbf{L}^{\top,-1}\Big)\epsilon\Bigg] \tag{57}$$

$$= \mathbb{E}_{\mathbf{y}_0}\mathbb{E}_{\epsilon}\Bigg[\epsilon^\top\Big(\mathbf{L}^{-1}g^2_\phi(s)\mathbf{L}^{\top,-1}\Big)\epsilon\Bigg] \tag{58}$$

$$= \mathbb{E}_{\epsilon}\Bigg[\epsilon^\top\Big(\mathbf{L}^{-1}g^2_\phi(s)\mathbf{L}^{\top,-1}\Big)\epsilon\Bigg] \tag{59}$$

$$= \text{Trace}\Bigg(\mathbf{L}^{-1}g^2_\phi(s)\mathbf{L}^{\top,-1}\Bigg) \tag{60}$$

F ELBOS IN STATIONARY PARAMETERIZATION

We use the stationary parmeterization described in appendix B. We now specialize the ELBO to the linear stationary parameterization.

Recall $f_\phi(\mathbf{y}, s) = -[\mathbf{Q}_\phi(s) + \mathbf{D}_\phi(s)]\mathbf{y}$. Recall $g_\phi(s) = \sqrt{2\mathbf{D}_\phi(s)}$ We have $g_\phi^2(s) = 2\mathbf{D}_\phi(s)$. We can write the MDM ISM ELBO as

$$\log p_\theta(x; \theta, \phi, \gamma)^{(ism)} \geq \mathbb{E}_{v \sim q_\gamma}\left[\mathbb{E}_{s \sim \text{Unif}(0,T)}\left[\ell_s^{(dsm)}\right] + \ell_T + \ell_q\right] \tag{61}$$

where

$$\begin{aligned}
\ell_{s_\theta} &= -\frac{1}{2}\|s_\theta(\mathbf{y}_s, s)\|^2_{\underbrace{2\mathbf{D}_\phi(s)}_{g_\phi^2}} \\
\ell_{\text{div-fgs}} &= \nabla_{\mathbf{y}_s} \cdot \Big[\underbrace{-[\mathbf{Q}_\phi(s) + \mathbf{D}_\phi(s)]\mathbf{y}_s}_{f_\phi} - \underbrace{2\mathbf{D}_\phi(s)}_{g_\phi^2} s_\theta(\mathbf{y}_s, s)\Big] \\
\ell_s^{\text{ism}} &= \mathbb{E}_{\underbrace{q_{\phi,s,(x,v)}}_{\text{depends on } \mathbf{Q},\mathbf{D}}}\Big[\ell_{s_\theta} + \ell_{\text{div-fgs}}\Big] \\
\ell_T &= \mathbb{E}_{\underbrace{q_{\phi,T},(x,v)}_{\text{depends on } \mathbf{Q},\mathbf{D}}}\Big[\log \pi_\theta(\mathbf{y}_T)\Big] \\
\ell_q &= -\log q_\gamma(v|x)
\end{aligned} \tag{62}$$

For the DSM form,

$$\log p_\theta(x; \theta, \phi, \gamma)^{(dsm)} \geq \mathbb{E}_{v \sim q_\gamma}\left[\mathbb{E}_{s \sim \text{Unif}(0,T)}\left[\ell_s^{(dsm)}\right] + \ell_T + \ell_q\right] \tag{63}$$

where

$$\begin{aligned}
\ell_{\text{div-f}} &= \nabla_{\mathbf{y}_s} \cdot \underbrace{-[\mathbf{Q}_\phi(s) + \mathbf{D}_\phi(s)]\mathbf{y}_s}_{f_\phi} \\
\ell_{\text{fwd-score}} &= \frac{1}{2}\Big|\Big|\underbrace{\nabla_{\mathbf{y}_s} \log q_\phi(\mathbf{y}_s|\mathbf{y}_0)}_{\text{depends on } \mathbf{Q},\mathbf{D}}\|^2_{\underbrace{2\mathbf{D}_\phi(s)}_{g_\phi^2}} \\
\ell_{\text{neg-scorediff}} &= -\frac{1}{2}\|s_\theta(\mathbf{y}_s, s) - \underbrace{\nabla_{\mathbf{y}_s} \log q_\phi(\mathbf{y}_s|\mathbf{y}_0)}_{\text{depends on } \mathbf{Q},\mathbf{D}}\|^2_{\underbrace{2\mathbf{D}_\phi(s)}_{g_\phi^2}} \\
\ell_s^{(dsm)} &= \mathbb{E}_{\underbrace{q_{\phi,s,(x,v)}}_{\text{depends on } \mathbf{Q},\mathbf{D}}}\Big[\ell_{\text{neg-scorediff}} + \ell_{\text{fwd-score}} + \ell_{\text{div-f}}\Big]
\end{aligned}$$

G ALGORITHMS

G.1 GENERIC TRANSITION KERNEL

Algorithm 2 Get transition distribution $\mathbf{y}_s|x$

Input: data x. time s. $\mathbf{A}, g$.
compute: $\mathbf{A}(s)$ and $g(s)$
compute: $\mathbf{M}_s = \int_0^s \mathbf{A}(t)dt$ (integrated drift)
compute: $\mathbf{N}_s = \int_0^s g^2(t)dt$ (integrated diffusions squared)
compute: $\gamma_{s|0} = \exp\left(\mathbf{M}_s\right)$ (mean coefficient)
set: $\mathbf{y}_0 = [x, 0_1, \ldots, 0_{K-1}]$, $\mathbf{\Sigma}_{0,zz} = \mathbf{0}$, and $\mathbf{\Sigma}_{0,zv}, \mathbf{\Sigma}_{0,vv}$ to chosen initial distribution
compute: $\mathbf{m}_{s|0} = \gamma_{s|0}\mathbf{y}_0$ (**mean**)
compute:

$$\begin{pmatrix}\mathbf{C}_s\\ \mathbf{H}_s\end{pmatrix} = \exp\left[\begin{pmatrix}\mathbf{M}_s & \mathbf{N}_s\\ \mathbf{0} & -\mathbf{M}_s^\top\end{pmatrix}\right]\begin{pmatrix}\mathbf{\Sigma}_0\\ \mathbf{I}\end{pmatrix} \quad \text{(ingredients for cov.)} \tag{64}$$

compute: $\mathbf{\Sigma}_{s|0} = \mathbf{C}_s(\mathbf{H}_s)^{-1}$ (**cov.**)
Output: $\mathcal{N}(\mathbf{m}_{s|0}, \mathbf{\Sigma}_{s|0})$

G.2 TRANSITIONS WITH Q, D

Current param matrices $\tilde{\mathbf{Q}}_\phi, \tilde{\mathbf{D}}_\phi$ and along with fixed time-in scalar-out functions $b_q(s), b_d(s)$ and their known integrals $B_q(s), B_d(s)$. $q_\gamma(v_0|z_0 = x)$ taken to be parameterless for now $v_0 \sim \mathcal{N}(0, I)$. Model params are s_θ fixed π_θ.

Algorithm 3 Get $\mathbf{Q}, \mathbf{D}$ and their integrated terms $\mathbf{M}, \mathbf{N}$

Input: time s and current params ϕ
compute: $[b_q]_s = \int_0^s b_q(\nu)d\nu$ using known integral $B_q(s) - B_q(0)$
compute: $[b_d]_s = \int_0^s b_d(\nu)d\nu$ using known integral $B_d(s) - B_d(0)$.
compute: $[\mathbf{Q}_\phi]_s = [b_q]_s \cdot \left[\tilde{\mathbf{Q}}_\phi - \tilde{\mathbf{Q}}_\phi^\top\right]$ for current params $\tilde{\mathbf{Q}}_\phi$.
compute: $[\mathbf{D}_\phi]_s = [b_d]_s \cdot \left[\tilde{\mathbf{D}}_\phi\tilde{\mathbf{D}}_\phi^\top\right]$ for current params $\tilde{\mathbf{D}}_\phi$.
compute: $\mathbf{M}_s = -([\mathbf{Q}_\phi]_s + [\mathbf{D}_\phi]_s)$ ($\mathbf{M}$ just a variable name)
compute: $\mathbf{N}_s = [2\mathbf{D}_\phi]_s = 2 \cdot [\mathbf{D}_\phi]_s$ ($\mathbf{N}$ just a variable name)
compute: $\mathbf{Q}_s = b_q(s) \cdot \left[\tilde{\mathbf{Q}}_\phi - \tilde{\mathbf{Q}}_\phi^\top\right]$ (not integrated)
compute: $\mathbf{D}_s = b_d(s) \cdot \left[\tilde{\mathbf{D}}_\phi\tilde{\mathbf{D}}_\phi^\top\right]$ (not integrated)
compute: $A_s = -[\mathbf{Q}_s + \mathbf{D}_s]$ (drift coef.)
compute: $g_s^2 = 2\mathbf{D}_s$ (diffusion coef. squared)
Output: $\mathbf{A}_s, g_s^2, \mathbf{M}_s, \mathbf{N}_s$

G.3 ELBO ALGORITHMS

Algorithm 4 Get transition distributions

Input: Sample $\mathbf{y}_0 = (x, v)$ and time s. Current params ϕ
set: $\mathbf{A}_s, g_s^2, \mathbf{M}_s, \mathbf{N}_s \leftarrow$ algorithm 3
compute: $\mathbf{m}_{s|0} = \exp\Big(\mathbf{M}_s\Big)\mathbf{y}_0$ (transition mean)
compute: ingredients for transition cov. matrix:

$$\begin{pmatrix}\mathbf{C}_s\\ \mathbf{H}_s\end{pmatrix} = \exp\left[\begin{pmatrix}\mathbf{M}_s & \mathbf{N}_s\\ \mathbf{0} & -\mathbf{M}_s^\top\end{pmatrix}\right]\begin{pmatrix}\mathbf{0}\\ \mathbf{I}\end{pmatrix} \tag{65}$$

compute: $\mathbf{\Sigma}_{s|0} = \mathbf{C}_s(\mathbf{H}_s)^{-1}$ (transition cov).
instantiate: $q_{\phi,s,(x,v)} = q_\phi(\mathbf{y}_s|\mathbf{y}_0) = \mathcal{N}(\mathbf{m}_{s|0}, \mathbf{\Sigma}_{s|0})$.
Output: $q_{\phi,s,(x,v)}, A_s, g_s^2$

Algorithm 5 Compute ELBO with ism or dsm

input: Data point x and current params θ, ϕ, γ
draw: an aux. sample $v \sim q_\gamma(v|x)$
draw: a sample $s \sim \text{Unif}(0, T)$
set: $\mathbf{y}_0 = (x, v)$
set: $q_{\phi,s,\mathbf{y}_0}, A_s, g_s^2 \leftarrow$ algorithm 4 called on $\mathbf{y}_0, s, \phi$
draw: $\mathbf{y}_s \sim q_{\phi,s,\mathbf{y}_0}$
compute: ℓ_s with dsm(s) (algorithm 6) or ism(s) (algorithm 7) on $\mathbf{y}_s, \theta, A_s, g_s^2, q_{\phi,s,\mathbf{y}_0}$
set: $q_{\phi,T,\mathbf{y}_0}, __, __ \leftarrow$ algorithm 4 called on $\mathbf{y}_0, T, \phi$
draw: $\mathbf{y}_T \sim q_{\phi,T,\mathbf{y}_0}$
output: $\ell_s + \log \pi_\theta(\mathbf{y}_T) - \log q_\gamma(v)$

Algorithm 6 Compute dsm(s)

input: $\mathbf{y}_s$, θ, A_s, g_s^2, $q_{\phi,s,\mathbf{y}_0}$.
compute: fwd-score $= \nabla_{\mathbf{y}_s} \log q_\phi(\mathbf{y}_s|\mathbf{y}_0)$
compute: model-score $= s_\theta(\mathbf{y}_s, s)$
compute: fwd-score-term $= \frac{1}{2}(\text{fwd-score})^\top g_s^2(\text{fwd-score})$
compute: score-diff $=$ model-score $-$ fwd-score
compute: diff-term $= -\frac{1}{2}\text{score-diff}^\top g_s^2\text{score-diff}$
compute: div-f $= \nabla_{\mathbf{y}_s} \cdot A_s\mathbf{y}_s$
output: dsm(s) $=$ fwd-score-term $+$ diff-term $+$ div-f

Algorithm 7 Compute ism(s)

input: $\mathbf{y}_s$, θ, A_s, g_s^2, $q_{\phi,s,\mathbf{y}_0}$.
compute: model-score $= s_\theta(\mathbf{y}_s, s)$
compute: score-term $= -\frac{1}{2}\text{model-score}^\top g_s^2\text{model-score}$
compute: div-gs $= \nabla_{\mathbf{y}_s} \cdot g_s^2 s_\theta(\mathbf{y}_s, s)$
compute: div-f $= \nabla_{\mathbf{y}_s} \cdot A_s\mathbf{y}_s$
compute: div-term $= -$div-gs $+$ div-f
output: ism(s) $=$ score-term $+$ div-term

H LIKELIHOOD AT TIME 0 USING TWEEDIE'S FORMULA

The integral term in the ELBO and its gradients have infinite variance at time 0. Following Sohl-Dickstein et al. (2015) and Song et al. (2021), we integrate the ELBO from $[\epsilon, T]$ rather than $[0, T]$. To make sure we still have a valid lower bound, we introduce a likelihood at time ϵ. The easiest way to think about this is that we have a model $p(u_0, u_\epsilon) = p(u_0|u_\epsilon)p(u_\epsilon)$ where the latent variable u_ϵ is modeled by a diffusion model. For this latent, the diffusion ELBO with integral term integrated $[\epsilon, T]$ is a proper lower bound. This is akin to lower-bounding the log prior term in variational inference. For the likelihood, we turn to Tweedie's formula to choose a $p_\theta(\mathbf{u}_0|\mathbf{u}_\epsilon)$ that matches closely to the inference's distributions $q(\mathbf{u}_0|\mathbf{u}_\epsilon)$.

H.1 RE-DERIVE THE ELBO WITH A SEPARATE VARIABLE $\mathbf{u}_\epsilon$

$$\log p(u_t) \geq \mathbb{E}_{q(u_T|u_\epsilon)}[\log \pi(u_T)] - \int_\epsilon^T \mathbb{E}_{q(u_t|u_\epsilon)}\Big[\nabla g^2 s_\theta + .5||s_\theta||^2_{g^2(s)} - \nabla \cdot f\Big] dt \tag{66}$$

$$\begin{aligned}
\log p(x) &= \log \int_{v0} p(x_0|v_0)p(v_0)dv_0 \\
&= \log \int_{v0} p(u_0 = [x, v_0])dv_0 \\
&= \log \int_{v0,u_\epsilon} p(u_0 = [x, v_0], u_\epsilon)du_\epsilon dv_0 \\
&= \log \int_{v0,u_\epsilon} p(u_0 = [x, v_0]|u_\epsilon)p(u_\epsilon)du_\epsilon dv_0 \\
&= \log \int_{v0,u_\epsilon} \frac{q(v_0|x)}{q(v_0|x)} \frac{q(u_\epsilon|u_0 = [x, v_0])}{q(u_\epsilon|u_0 = [x, v_0])} p(u_0 = [x, v_0]|u_\epsilon)p(u_\epsilon)du_\epsilon dv_0 \\
&= \log E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\left[\frac{1}{q(v_0|x)} \frac{1}{q(u_\epsilon|u_0 = [x, v_0])} p(u_0 = [x, v_0]|u_\epsilon)p(u_\epsilon)\right] \\
&\geq E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\Big[\log p(u_0 = [x, v_0]|u_\epsilon) + \log p(u_\epsilon) - \log q(v_0|x) - \log q(u_\epsilon|u_0 = [x, v_0])\Big] \\
&= E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\left[\left(\log p(u_0 = [x, v_0]|u_\epsilon) - \log q(u_\epsilon|u_0 = [x, v_0])\right) + \log p(u_\epsilon) - \log q(v_0|x)\right] \\
&= E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\left[\left(\log p(u_0 = [x, v_0]|u_\epsilon) - \log q(u_\epsilon|u_0 = [x, v_0])\right) - \log q(v_0|x)\right] \\
&\quad + E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\Big[\log p(u_\epsilon)\Big] \\
&\geq E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\left[\left(\log p(u_0 = [x, v_0]|u_\epsilon) - \log q(u_\epsilon|u_0 = [x, v_0])\right) - \log q(v_0|x)\right] \\
&\quad + E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\left[E_{q(u_T|u_\epsilon)}[\log \pi(u_T)] - \int_\epsilon^T \mathbb{E}_{q(u_t|u_\epsilon)}\Big[\nabla \cdot g^2 s_\theta + .5||s_\theta||^2_{g^2} - \nabla \cdot f\Big] dt\right] \\
&= E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\left[\left(\log p(u_0 = [x, v_0]|u_\epsilon) - \log q(u_\epsilon|u_0 = [x, v_0])\right) - \log q(v_0|x)\right] \\
&\quad + E_{q(v_0,u_\epsilon,u_T|x)}\Big[\log \pi(u_T)\Big] - \int_\epsilon^T \left[\mathbb{E}_{q(v_0,u_t,u_\epsilon|x)}\Big[\nabla \cdot g^2 s_\theta + .5||s_\theta||^2_{g^2} - \nabla \cdot f\Big] dt\right] \\
&= E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\left[\left(\log p(u_0 = [x, v_0]|u_\epsilon) - \log q(u_\epsilon|u_0 = [x, v_0])\right) - \log q(v_0|x)\right] \\
&\quad + E_{q(u_T|x)q(u_\epsilon,v_0|u_T,x)}\Big[\log \pi(u_T)\Big] - \int_\epsilon^T \left[\mathbb{E}_{q(u_t|x)q(u_\epsilon,v_0|u_t,x])}\Big[\nabla \cdot g^2 s_\theta + .5||s_\theta||^2_{g^2} - \nabla \cdot f\Big] dt\right] \\
&= E_{q(v_0|x)q(u_\epsilon|u_0=[x,v_0])}\left[\left(\log p(u_0 = [x, v_0]|u_\epsilon) - \log q(u_\epsilon|u_0 = [x, v_0])\right) - \log q(v_0|x)\right] \\
&\quad + E_{q(u_T|x)}\Big[\log \pi(u_T)\Big] - \int_\epsilon^T \left[\mathbb{E}_{q(u_t|x)}\Big[\nabla \cdot g^2 s_\theta + .5||s_\theta||^2_{g^2} - \nabla \cdot f\Big] dt\right]
\end{aligned}$$

Given $\mathbf{u}_0 \sim q_0(\mathbf{u}_0)$ and $\mathbf{u}_\epsilon \sim N(\mathbf{u}_\epsilon \mid A\mathbf{u}_0, \Sigma)$, we use Tweedie's formula to get $\mathbb{E}[\mathbf{u}_0 \mid \mathbf{u}_\epsilon]$ under q, and then set the mean of $p(\mathbf{u}_0|\mathbf{u}_\epsilon)$ that function. Here $\mathbf{u}_0 = [\mathbf{x}_0, \mathbf{v}_0]$ where $\mathbf{x}_0 \sim q_{\text{data}}$.

Let η be the natural parameter for the multivariate Gaussian likelihood, with known covariance and unknown mean. The parameter η is defined as $\Sigma^{-1}\mu = \Sigma^{-1}A\mathbf{u}_0$. Then, Tweedie's formula states that

$$\mathbb{E}[\eta \mid \mathbf{u}_\epsilon] = \nabla_{\mathbf{u}_\epsilon} l(\mathbf{u}_\epsilon) - \nabla_{\mathbf{u}_\epsilon} l_0(\mathbf{u}_\epsilon)$$

where $l(\mathbf{u}_\epsilon) = \log q(\mathbf{u}_\epsilon)$, and $\nabla_{\mathbf{u}_\epsilon} l(\mathbf{u}_\epsilon) = s_\theta(\mathbf{u}_\epsilon, \epsilon)$ (again, like above, assuming score model correct at time ϵ), and l_0 is the log of the base distribution defined in the exponential family parameterization.

The base distribution is a multivariate Gaussian with mean 0 and covariance Σ, therefore $\nabla_{\mathbf{u}_\epsilon} l_0(\mathbf{u}_\epsilon) = -\Sigma^{-1}\mathbf{u}_\epsilon$,

$$\mathbb{E}[\eta \mid \mathbf{u}_\epsilon] = s_\theta(\mathbf{u}_\epsilon, \epsilon) + \Sigma^{-1}\mathbf{u}_\epsilon.$$

Now, to derive the conditional mean of $\mathbf{u}_0$ given $\mathbf{u}_\epsilon$, we use the linearity of conditional expectations

$$\begin{aligned}
\mathbb{E}[\mathbf{u}_0 \mid \mathbf{u}_\epsilon] &= \mathbb{E}[A^{-1}\Sigma\eta|\mathbf{u}_\epsilon] \\
&= A^{-1}\Sigma\mathbb{E}[\eta \mid \mathbf{u}_\epsilon] \\
&= A^{-1}\Sigma\left(s_\theta(\mathbf{u}_\epsilon, \epsilon) + \Sigma^{-1}\mathbf{u}_\epsilon\right) \\
&= A^{-1}\Sigma s_\theta(\mathbf{u}_\epsilon, \epsilon) + A^{-1}\mathbf{u}_\epsilon
\end{aligned}$$

where we use the fact that $\mathbf{u}_0 = A^{-1}\Sigma\eta$ which holds because $\eta = \Sigma^{-1}A\mathbf{u}_0$

For the variance, we use the following relation $\mathbf{u}_\epsilon = A\mathbf{u}_0 + \sqrt{\Sigma}\epsilon$, which implies that

$$\begin{aligned}
\mathbf{u}_0 &= A^{-1}\mathbf{u}_\epsilon - A^{-1}\sqrt{\Sigma}\epsilon \\
\text{Var}\left(\mathbf{u}_0 \mid \mathbf{u}_\epsilon\right) &= A^{-1}\Sigma A^{-T}.
\end{aligned}$$

Therefore, for the posterior distribution $r(\mathbf{u}_0 \mid \mathbf{u}_\epsilon)$ we choose

$$\begin{aligned}
\mu_r &= A^{-1}\Sigma s_\theta(\mathbf{u}_\epsilon, \epsilon) + A^{-1}\mathbf{u}_\epsilon \\
\Sigma_r &= A^{-1}\Sigma A^{-T}
\end{aligned}$$

Here, A, Σ are the mean coefficient and covariance derived from the transition kernel at time ϵ.

We use these means and covariances of $q(\mathbf{u}_0|\mathbf{u}_\epsilon)$ to set the mean and covariance of the likelihood $p_\theta(\mathbf{u}_0|\mathbf{u}_\epsilon)$.