Calibrating Generative Models

Henry Smith [1] **Brian L. Trippe** [1]

Abstract

Generative models frequently suffer miscalibration, wherein class probabilities and other statistics of the sampling distribution deviate from desired values. We frame calibration as a constrained optimization problem and seek the minimally perturbed model (in Kullback-Leibler divergence) satisfying calibration constraints. To address the intractability of the hard constraint, we introduce two surrogate objectives: (1) the relaxed loss, which replaces the constraint with a miscalibration penalty, and (2) the reward loss, which we derive as a divergence to a known, approximate solution. We show how to minimize these losses for neural-SDE models and find they solve synthetic calibration problems to high precision. Lastly, we demonstrate the practicality of the approach by calibrating a 15M-parameter protein structure diffusion model to match the distribution of secondary structure composition of natural proteins.

1. Introduction

Generative models commonly produce samples whose statistics deviate systematically from desired values. Such *miscalibration* occurs across a range of domains. In protein structure modeling, for instance, protein design models sample alpha-helical and beta-strand structural motifs at frequencies atypical of proteins in nature (Lu et al., 2025). Language models disproportionately sample words like "delve" and "crucial" (Kobak et al., 2024). And protein diffusion models trained to approximate Boltzmann distributions fail to produce structures in different states with probabilities that are consistent with experimental measurements (Lewis et al., 2024). Similar phenomena exist in vision models, where image generators disproportionately render clocks showing 10:10 AM or PM.

[1]Department of Statistics, Stanford University, Palo Alto, USA. Correspondence to: Henry Smith <smithhd@stanford.edu>, Brian L. Trippe <btrippe@stanford.edu>.

Proceedings of the Workshop on Generative AI for Biology at the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

These calibration errors can arise from many sources: dataset imbalances, model capacity limitations, suboptimal training dynamics, or post-hoc adjustments such as low-temperature sampling or preference fine-tuning. While reward-based fine-tuning methods such as direct preference optimization (Schulman et al., 2017; Fan et al., 2023; Wallace et al., 2024; Black et al., 2024) and adjoint matching (Domingo-Enrich et al., 2025) allow practitioners to incorporate preferences that may be specified on the level of individual samples, calibration is a hard constraint specified at the distribution level.

To address this gap, we frame calibration as a constrained optimization problem: find the distribution closest in KL divergence to the base model that satisfies a set of expectation constraints. We introduce two algorithms—**CGM-relax** ("calibrating generative models") and **CGM-reward**—that approximately solve the calibration problem by stochastic optimization. These algorithms apply to general moment constraints and are compatible with high-dimensional generative models.

2. Problem Statement

Consider a pre-trained "base" generative model $p_{\theta_{\text{base}}}(\boldsymbol{x})$ with parameters θ_{base}, a statistic $\boldsymbol{h}(\boldsymbol{x})$, and an expectation value desired for the statistic $\boldsymbol{h}^*$. We say $p_{\theta_{\text{base}}}$ is *calibrated* if $\mathbb{E}_{p_{\theta_{\text{base}}}}[\boldsymbol{h}(\boldsymbol{x})] = \boldsymbol{h}^*$ and *miscalibrated* if $\mathbb{E}_{p_{\theta_{\text{base}}}}[\boldsymbol{h}(\boldsymbol{x})] \neq \boldsymbol{h}^*$. In the case that $p_{\theta_{\text{base}}}$ is miscalibrated, our goal is to fine-tune its parameters θ_{base} to some θ such that p_θ is calibrated.

For example, if $\boldsymbol{h}(\boldsymbol{x}) = 1$ for $\boldsymbol{x}$ belonging to a particular class and $\boldsymbol{h}(\boldsymbol{x}) = 0$ otherwise, then the value $\boldsymbol{h}^*$ corresponds to the probability that $\boldsymbol{x}$ belongs to that class. And if $\boldsymbol{h}(\boldsymbol{x}) = [t(\boldsymbol{x}), t(\boldsymbol{x})^2]$ for a scalar statistic $t(\boldsymbol{x})$, then the choice $\boldsymbol{h}^* = [\mu, \sigma^2 + \mu^2]$ corresponds to the constraint that $t(\boldsymbol{x})$ has mean μ and variance σ^2.

For a given $\boldsymbol{h}(\cdot)$ and $\boldsymbol{h}^*$, many calibrated models may exist. Provided a calibrated model exists, we seek the one that is minimally perturbed from the base model,

$$p_{\theta^*} := \underset{p_\theta \text{ s.t. } \mathbb{E}_{p_\theta}[\boldsymbol{h}(\boldsymbol{x})]=\boldsymbol{h}^*}{\arg\min} \mathrm{D}_{\mathrm{KL}}\left(p_\theta \parallel p_{\theta_{\text{base}}}\right). \tag{1}$$

Equation (1) realizes the notion of the minimally perturbed model as the one that is closest in the Kullback-Leibler (KL)

divergence $\mathrm{D}_{\mathrm{KL}}\left(p' \parallel p\right) = \mathbb{E}_{p'}[\log p'(\boldsymbol{x})/p(\boldsymbol{x})]$, where p' is assumed to have a probability density with respect to p.

Out of many possible notions of distance we focus on D_{KL} because it is simple and, as we will see, can be computationally tractable even for some complex generative models.

3. Calibrating Generative Models with CGM-relax and CGM-reward

The calibration problem is challenging to solve for non-trivial generative models for two principal reasons. First, both the objective and the calibration constraint in equation (1) are defined by expectations that are intractable to compute. Second, even if these expectations could be computed, neither the objective function nor set of parameters θ that satisfy the calibration constraint is, in general, convex.

To address the intractability of the constraint, we propose two alternative objectives whose *unconstrained* optima approximate the solution to (1). These objectives still involve expectations under p_θ and are non-convex; however, we demonstrate that one can obtain unbiased estimates for both the objectives themselves as well as their gradients. As a result, they are amenable to stochastic optimization.

We call our algorithms optimizing the two surrogate loss functions CGM-relax and CGM-reward (Algorithms 1 and 2, respectively). These algorithms require only that one can draw samples from p_θ and that $p_\theta(\boldsymbol{x})$ is computable and has computable gradients with respect to θ.

3.1. The Relaxed Loss

The CGM-relax loss avoids the intractability of imposing the calibration constraint exactly by replacing it with a constraint violation penalty

$$\mathcal{L}^{\text{relax}}(\theta) := \underbrace{\mathrm{D}_{\mathrm{KL}}\left(p_\theta \parallel p_{\theta_{\text{base}}}\right)}_{\mathcal{L}^{\text{KL}}} + \lambda \underbrace{\| \mathbb{E}_{p_\theta}[\boldsymbol{h}(\boldsymbol{x})] - \boldsymbol{h}^* \|^2}_{\mathcal{L}^{\text{viol}}}, \quad (2)$$

where $\lambda \in \mathbb{R}_+$ is a hyperparameter that controls the tradeoff between satisfying the calibration constraint and minimizing the KL divergence. In the limit as $\lambda \to \infty$, we can expect the minimizer of (2) to approach that of (1).

Suppose we have M independent samples $\{\boldsymbol{x}_m\}_{m=1}^M$ from the generative model p_θ. In order to construct an unbiased estimate for $\mathcal{L}^{\text{relax}}$, we separately construct unbiased estimates for $\mathcal{L}^{\text{KL}}$, the KL divergence, and for $\mathcal{L}^{\text{viol}}$, the squared norm difference between the actual and target expectation of $\boldsymbol{h}$. For the KL, we propose a Monte Carlo estimator

$$\widehat{\mathcal{L}}^{\text{KL}} := \frac{1}{M} \sum_{m=1}^{M} \log \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)}.$$

And for the squared norm difference, we use the bias-variance decomposition to construct the estimator

$$\widehat{\mathcal{L}}^{\text{viol}} := \left\| \frac{1}{M} \sum_{m=1}^{M} \tilde{\boldsymbol{h}}_m \right\|^2 - \frac{1}{M(M-1)} \sum_{m=1}^{M} \left\| \tilde{\boldsymbol{h}}_m - \frac{1}{M} \sum_{m'=1}^{M} \tilde{\boldsymbol{h}}_{m'} \right\|^2, \quad (3)$$

where we have defined $\tilde{\boldsymbol{h}}_m = \boldsymbol{h}(\boldsymbol{x}_m) - \boldsymbol{h}^*$ for $m = 1, \ldots, M$. Combining these estimators yields our overall estimator for the relaxed objective,

$$\widehat{\mathcal{L}}^{\text{relax}} = \widehat{\mathcal{L}}^{\text{KL}} + \lambda \widehat{\mathcal{L}}^{\text{viol}}. \quad (4)$$

We prove in Appendix A.1 that $\widehat{\mathcal{L}}^{\text{relax}}$ is unbiased for $\mathcal{L}^{\text{relax}}$.

3.2. The Reward Loss

The CGM-reward loss avoids the intractability of imposing the calibration constraint exactly by leveraging a connection between the calibration problem (1) and the *maximum entropy problem* (Kullback, 1959; Kullback & Khairat, 1966; Csiszár, 1975). We first introduce the maximum entropy problem. We then show how to approximate its solution with samples from $p_{\theta_{\text{base}}}$. Lastly, we propose the reward loss $\mathcal{L}^{\text{reward}}$ as a divergence to this approximate solution and describe connections to reward fine-tuning.

Maximum entropy problem. The maximum entropy problem solves

$$\underset{p \in \mathcal{P}(p_{\theta_{\text{base}}}) \text{ s.t. } \mathbb{E}_p[\boldsymbol{h}(\boldsymbol{x})] = \boldsymbol{h}^*}{\arg\min} \mathrm{D}_{\mathrm{KL}}\left(p \parallel p_{\theta_{\text{base}}}\right), \quad (5)$$

where $\mathcal{P}(p)$ is the collection of probability distributions that have a density with respect to p. The calibration problem and the maximum entropy problem differ only in their domains: the domain of the calibration problem is generative models p_θ in the same parametric class as $p_{\theta_{\text{base}}}$, rather than the nonparametric set $\mathcal{P}(p_{\theta_{\text{base}}})$. Despite this difference we obtain an alternative objective by considering the solution to (5). The following theorem characterizes this solution.

Theorem 3.1. *Suppose there exists $\boldsymbol{\alpha}^*$ such that*

$$p_{\boldsymbol{\alpha}^*}(\boldsymbol{x}) \propto p_{\theta_{\text{base}}}(\boldsymbol{x}) \exp\left\{r_{\boldsymbol{\alpha}^*}(\boldsymbol{x})\right\}, \quad r_{\boldsymbol{\alpha}}(\boldsymbol{x}) := \boldsymbol{\alpha}^{\mathsf{T}} \boldsymbol{h}(\boldsymbol{x}) \quad (6)$$

satisfies $\mathbb{E}_{p_{\boldsymbol{\alpha}^*}}[\boldsymbol{h}(\boldsymbol{x})] = \boldsymbol{h}^*$. Then, under conditions, $p_{\boldsymbol{\alpha}^*}$ is the unique solution to problem (5).

Appendix B gives conditions for and a proof of Theorem 3.1 adapted from Kullback & Khairat (1966). For the remainder of our discussion, we will assume that an $\boldsymbol{\alpha}^*$ satisfying the conditions of Theorem 3.1 exists; Appendix B.1 details conditions that guarantee this existence.

The domain of calibration problem (1) may not contain $p_{\boldsymbol{\alpha}^*}$. However, if the class of generative models p_θ is sufficiently

Algorithm 1 CGM-relax fine-tuning

Require: $p_{\theta_{\text{base}}}, h(\cdot), \boldsymbol{h}^*, M$, and λ
▷ Initialize and optimize
$p_\theta \leftarrow p_{\theta_{\text{base}}}$
while not converged **do**
  ▷ Sample and compute weights
  $\boldsymbol{x}_1, \dots, \boldsymbol{x}_M \overset{i.i.d.}{\sim} p_{\texttt{stop-grad}(\theta)}$
  $w_m \leftarrow p_\theta(\boldsymbol{x}_m)/p_{\texttt{stop-grad}(\theta)}(\boldsymbol{x}_m)$
  ▷ Kullback-Leibler loss, with LOO baseline
  $l_m \leftarrow \log p_{\texttt{stop-grad}(\theta)}(\boldsymbol{x}_m)/p_{\theta_{\text{base}}}(\boldsymbol{x}_m)$
  $l_m^{\text{LOO}} \leftarrow l_m - \frac{1}{M-1}\sum_{m' \neq m} l_{m'}$
  $\widehat{\mathcal{L}}^{\text{KL}} \leftarrow \frac{1}{M}\sum w_m l_m^{\text{LOO}}$
  ▷ Constraint violation and bias correction
  $\boldsymbol{h}_m \leftarrow w_m(\boldsymbol{h}(\boldsymbol{x}_m) - \boldsymbol{h}^*)$
  $\widehat{\mathcal{L}}^{\text{viol}} \leftarrow \|\frac{1}{M}\sum \boldsymbol{h}_m\|^2 - \frac{1}{M-1}\widehat{\text{Var}}[\boldsymbol{h}_{1:M}]$,
    where $\widehat{\text{Var}}[\boldsymbol{h}_{1:M}] = \frac{1}{M}\sum \|\boldsymbol{h}_m - \frac{1}{M}\sum \boldsymbol{h}_{m'}\|^2$
  ▷ Total loss and update
  $\widehat{\mathcal{L}}^{\text{relax}} = \widehat{\mathcal{L}}^{\text{KL}} + \lambda\widehat{\mathcal{L}}^{\text{viol}}$
  $\theta \leftarrow \text{gradient-step}(\theta, \nabla_\theta \widehat{\mathcal{L}}^{\text{relax}})$

expressive, then the optimum of the calibration problem p_{θ^*} will be close to $p_{\boldsymbol{\alpha}^*}$. This observation suggests a second way to remove the constraint in equation (1): fine-tune p_θ to minimize a divergence to $p_{\boldsymbol{\alpha}^*}$.

Estimating $p_{\boldsymbol{\alpha}^*}$. The idea of minimizing a divergence to $p_{\boldsymbol{\alpha}^*}$ introduces a challenge: although we know the solution $p_{\boldsymbol{\alpha}^*}$ to the maximum entropy problem (5) exists, we do not know how to determine its parameters $\boldsymbol{\alpha}^*$. To address this challenge, we leverage a result from Wainwright et al. (2008, Chapter 3.6), which states that, under conditions, solving the maximum entropy problem (5) is equivalent to solving the problem

$$\arg\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log\left(\int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\}p_{\theta_{\text{base}}}(\boldsymbol{x})d\boldsymbol{x}\right). \quad (7)$$

In other words, by solving (7) one obtains the parameters $\boldsymbol{\alpha}^*$ of $r_{\boldsymbol{\alpha}}(\boldsymbol{x})$, which then determines the solution $p_{\boldsymbol{\alpha}^*}$ to the maximum entropy problem up to a normalizing constant.

One issue with solving equation (7) is that for the pre-trained generative models of interest, the second term involves an intractable integral. Tohme et al. (2023) discuss approximating this integral with numerical integration; however, this approach is infeasible when the dimension of $\boldsymbol{x}$ is large. We instead propose drawing M independent samples $\{\boldsymbol{x}_m\}_{m=1}^M$ from $p_{\theta_{\text{base}}}$ and replacing the integral with respect to $p_{\theta_{\text{base}}}$ by the integral with respect to the empirical distribution that places probability mass M^{-1} on each of the samples $\boldsymbol{x}_m$,

$$\widehat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log\left(\frac{1}{M}\sum_{m=1}^M \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x}_m)\}\right). \quad (8)$$

The problem (8) is concave, and when $\widehat{\boldsymbol{\alpha}}$ is well-defined (see Appendix B.1), it can be found by convex solvers. We

Algorithm 2 CGM-reward fine-tuning

Require: $p_{\theta_{\text{base}}}, h(\cdot), \boldsymbol{h}^*, M, N$
▷ Estimate $\boldsymbol{\alpha}$ for reward
$\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \overset{i.i.d.}{\sim} p_{\theta_{\text{base}}}$
$\widehat{\boldsymbol{\alpha}} \leftarrow \arg\max \boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log\frac{1}{N}\sum \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x}_n)\}$
▷ Initialize and optimize
$p_\theta \leftarrow p_{\theta_{\text{base}}}$
while not converged **do**
  ▷ Sample and compute weights
  $\boldsymbol{x}_1, \dots, \boldsymbol{x}_M \overset{iid}{\sim} p_{\texttt{stop-grad}(\theta)}$
  $w_m \leftarrow p_\theta(\boldsymbol{x}_m)/p_{\texttt{stop-grad}(\theta)}(\boldsymbol{x}_m)$
  ▷ Kullback-Leibler loss, with LOO baseline
  $l_m \leftarrow \log p_{\texttt{stop-grad}(\theta)}(\boldsymbol{x}_m)/p_{\theta_{\text{base}}}(\boldsymbol{x}_m)$
  $l_m^{\text{LOO}} \leftarrow l_m - \frac{1}{M-1}\sum_{m' \neq m} l_{m'}$
  $\widehat{\mathcal{L}}^{\text{KL}} \leftarrow \frac{1}{M}\sum w_m l_m^{\text{LOO}}$
  ▷ Negative reward loss, with LOO baseline
  $r_m^{\text{LOO}} \leftarrow r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_m) - \frac{1}{M-1}\sum_{m' \neq m} r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_{m'})$
  $\widehat{\mathcal{L}}^{\text{r}} \leftarrow -\frac{1}{M}\sum w_m r_m^{\text{LOO}}$
  ▷ Total loss and update
  $\widehat{\mathcal{L}}^{\text{reward}} = \widehat{\mathcal{L}}^{\text{KL}} + \widehat{\mathcal{L}}^{\text{r}}$
  $\theta \leftarrow \text{gradient-step}(\theta, \nabla_\theta \widehat{\mathcal{L}}^{\text{reward}})$

demonstrate in Appendix B.1 that $\widehat{\boldsymbol{\alpha}}$ converges to $\boldsymbol{\alpha}^*$ in the limit of many samples M.

$\mathcal{L}^{\text{reward}}$ and its estimation. With $\widehat{\boldsymbol{\alpha}}$ in hand, we formulate our second loss as a divergence to $p_{\widehat{\boldsymbol{\alpha}}}$. For simplicity and because it avoids the requirement to compute the normalizing constant of $p_{\widehat{\boldsymbol{\alpha}}}$, we again choose the KL divergence. In particular, we define the reward loss $\mathcal{L}^{\text{reward}}$ to be

$$\begin{aligned}\mathcal{L}^{\text{reward}}(\theta) &= \text{D}_{\text{KL}}\left(p_\theta \parallel p_{\widehat{\boldsymbol{\alpha}}}\right) \\ &= \underbrace{\mathbb{E}_{p_\theta}[\log p_\theta(\boldsymbol{x})/p_{\theta_{\text{base}}}(\boldsymbol{x})]}_{\mathcal{L}^{\text{KL}} = \text{D}_{\text{KL}}\left(p_\theta \parallel p_{\theta_{\text{base}}}\right)} + \underbrace{\mathbb{E}_{p_\theta}[-r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})]}_{\mathcal{L}^{\text{r}}} + C, \end{aligned} \quad (9)$$

where $C = \mathbb{E}_{p_{\theta_{\text{base}}}}[\exp\{r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\}]$ is a normalizing constant that does not depend on θ and so can be ignored.

We call $r_{\boldsymbol{\alpha}}(\boldsymbol{x})$ the *reward* and $\mathcal{L}^{\text{reward}}$ the reward loss because $\mathcal{L}^{\text{reward}}$ coincides with the objective of reward fine-tuning algorithms (e.g., Rafailov et al., 2023; Wallace et al., 2024; Uehara et al., 2024; Domingo-Enrich et al., 2025). The goal of reward fine-tuning is to fine-tune the base generative model $p_{\theta_{\text{base}}}$ to a tilted version of itself, where the tilt is determined by a so-called reward $r(\boldsymbol{x})$.

As with $\widehat{\mathcal{L}}^{\text{KL}}$ in Section 3.1, Monte Carlo sampling provides an unbiased estimate as

$$\widehat{\mathcal{L}}^{\text{reward}} := \frac{1}{M}\sum_{m=1}^M \left(\log\frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_m)\right).$$

3.3. Gradient Estimation

We next describe approaches to compute unbiased estimates for gradients of $\mathcal{L}^{\text{relax}}(\theta)$ and $\mathcal{L}^{\text{reward}}(\theta)$ with respect to θ that permit minimization of these losses by stochastic optimization. We first describe our general approach which is a generalization of the score-function gradient estimator. We then propose adjustments of the associated gradient estimators $\mathcal{L}^{\text{KL}}$ and $\mathcal{L}^{\text{r}}$ to reduce variance that we use in our experiments. Lastly, we present our estimator of $\nabla_\theta \mathcal{L}^{\text{viol}}(\theta)$. Appendix A.2 provides conditions for and proofs of the unbiasedness of the resulting gradient estimators.

Score-based gradient estimation. The primary challenge to computing gradients is the inability to directly exchange the order of the gradients and expectations taken with respect to θ; that is because $\nabla_\theta \mathcal{L}(\theta) = \nabla_\theta \mathbb{E}_{p_\theta}[f(\boldsymbol{x}, \theta)] \neq \mathbb{E}_{p_\theta}[\nabla_\theta f(\boldsymbol{x}, \theta)]$, $\nabla_\theta \mathcal{L}(\theta)$ can not in general be usefully approximated by $M^{-1} \sum \nabla_\theta f(\boldsymbol{x}_m, \theta)$ from samples $\boldsymbol{x}_m$ of p_θ. To address this challenge, we first observe that we can rewrite $\mathcal{L}(\theta) = \mathcal{L}(\theta, \theta')$ for

$$\mathcal{L}(\theta, \theta') = \mathbb{E}_{p_{\theta'}}\left[\frac{p_\theta(\boldsymbol{x})}{p_{\theta'}(\boldsymbol{x})} f(\boldsymbol{x}, \theta)\right], \tag{10}$$

where θ' is another arbitrary set of model parameters. Since the expectation in equation (10) does not depend on θ, we *can* approximate it with Monte Carlo samples from $p_{\theta'}$. Moreover, since the choice of θ' is arbitrary, we can choose $\theta'{=}\theta$, draw independent samples $\boldsymbol{x}_m$ from p_θ, and compute

$$\begin{aligned}&[\nabla_\theta \mathcal{L}(\theta, \theta')] \mid_{\theta'=\theta} \approx [\nabla_\theta \hat{\mathcal{L}}(\theta, \theta')] \mid_{\theta'=\theta} \\ =&\frac{1}{M} \sum \frac{1}{p_\theta(\boldsymbol{x}_m)} \nabla_\theta [p_\theta(\boldsymbol{x}_m) f(\boldsymbol{x}_m, \theta)].\end{aligned} \tag{11}$$

The density ratio $p_\theta(\boldsymbol{x}_m)/p_{\theta'}(\boldsymbol{x}_m)$ in equation (10) can be understood as the weights of an importance sampling estimate against target p_θ with proposal $p_{\theta'}$. Perhaps unintuitively, when $\theta'{=}\theta$ this ratio is necessarily equal to one by construction but its gradient is the "score" function $(\nabla_\theta p_\theta(\boldsymbol{x}_m))/p_\theta(\boldsymbol{x}_m) = \nabla \log p_\theta(\boldsymbol{x})$ which is non-trivial in general (Mohamed et al., 2020). Algorithms 1 and 2 each demonstrate an implementation that computes these weights with a copy of the parameters θ detached from the computational graph, which we denote by $\texttt{stop-grad}(\theta)$.

Variance reduction of $\nabla_\theta \widehat{\mathcal{L}}^{\text{KL}}$. In the case of the KL loss, for which $f(\boldsymbol{x}, \theta) = \log[p_\theta(\boldsymbol{x})/p_{\theta_{\text{base}}}(\boldsymbol{x})]$, an application of the product rule expands each term of equation (11) into

$$\begin{aligned}&\frac{1}{p_\theta(\boldsymbol{x}_m)} \nabla_\theta \left[p_\theta(\boldsymbol{x}_m) \log \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)}\right] = \\ &[\nabla_\theta \log p_\theta(\boldsymbol{x}_m)] \log \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)} + \underbrace{1 \cdot \nabla_\theta \log \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)}}_{=0 \text{ in expectation}}.\end{aligned}$$

While both terms contribute to the variance of $\nabla_\theta \widehat{\mathcal{L}}^{\text{KL}}(\theta, \theta')$, the second term has expectation zero for any θ and so can be dropped to reduce variance without sacrificing unbiasedness. The resulting estimate of $\nabla_\theta \mathcal{L}^{\text{KL}}(\theta)$ is

$$\frac{1}{M} \sum [\nabla_\theta \log p_\theta(\boldsymbol{x}_m)] \log \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)}.$$

This idea is adapted from Ranganath et al. (2014), who consider a similar score-based gradient estimate for the KL divergence in the context of variational Bayesian inference.

The integrands of $\mathcal{L}^{\text{r}}$ and $\mathcal{L}^{\text{viol}}$ in equations (2) and (9) do not depend on θ, and so the gradient in equation (11) involves only a single term.

Variance reduction with a baseline. As a second strategy to reduce the variance of the score-based gradient estimate, we subtract from each term in the gradient estimate (11) a *baseline* or *control variate*, which is a term that has expectation zero under p_θ and is correlated with each individual term in the estimate (Lavenberg & Welch, 1981; Ranganath et al., 2014; Mohamed et al., 2020). A simple choice of baseline is the product of the score of each sample and the leave-one-out (LOO) average of the evaluations of the integrand in equation (10) (Kool et al., 2019). In the case of the estimate of $\nabla_\theta \mathcal{L}^{\text{KL}}(\theta)$ we obtain

$$\frac{1}{M} \sum_{m=1}^{M} (\nabla_\theta \log p_\theta(\boldsymbol{x}_m)) \left(\log \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)} - \text{LOO}_m^{\text{KL}}\right)$$

$$\text{where} \quad \text{LOO}_m^{\text{KL}} := \frac{1}{M-1} \sum_{m' \neq m} \log \frac{p_\theta(\boldsymbol{x}_{m'})}{p_{\theta_{\text{base}}}(\boldsymbol{x}_{m'})}.$$

In the same way, we include a LOO baseline for $\nabla_\theta \mathcal{L}^{\text{r}}$.

Unbiased estimation of $\nabla_\theta \mathcal{L}^{\text{viol}}$. Equation (11) does not immediately apply to $\mathcal{L}^{\text{viol}}$ since $\mathcal{L}^{\text{viol}}$ cannot be expressed in the form $\mathbb{E}_{p_\theta}[f(\boldsymbol{x}, \theta)]$ for some function f. Instead, we begin with $\widehat{\mathcal{L}}^{\text{viol}}$ in equation (3) which depends on M samples and compute a gradient estimate as $[\nabla_\theta \widehat{\mathcal{L}}^{\text{viol}}(\theta, \theta')] \mid_{\theta'=\theta}$. Here,

$$\begin{aligned}\widehat{\mathcal{L}}^{\text{viol}}(\theta, \theta') :=& \left\| \frac{1}{M} \sum\nolimits_m w_m \tilde{\boldsymbol{h}}_m \right\|^2 - \\ &\frac{1}{M(M-1)} \sum\nolimits_m \left\| w_m \tilde{\boldsymbol{h}}_m - \frac{1}{M} \sum\nolimits_{m'} w_{m'} \tilde{\boldsymbol{h}}_{m'} \right\|^2,\end{aligned}$$

where $w_m = p_\theta(\boldsymbol{x}_m)/p_{\theta'}(\boldsymbol{x}_m)$ and $\tilde{\boldsymbol{h}}_m = \boldsymbol{h}(\boldsymbol{x}_m) - \boldsymbol{h}^*$. Similar to equation (11), $\widehat{\mathcal{L}}^{\text{viol}}(\theta, \theta')$ has the interpretation as an importance weighted estimate. We detail this interpretation in Appendix A.2 alongside a proof of unbiasedness.

3.4. CGM-relax and CGM-reward

Algorithms 1 and 2 provide pseudo-code for CGM-relax and CGM-reward, which approximately solve the calibration

problem (1) by optimizing $\mathcal{L}^{\text{relax}}$ and $\mathcal{L}^{\text{reward}}$, respectively. Both algorithms initialize model parameters to those of the base model, and use stochastic optimization with unbiased gradient estimates computed with samples from the current model iterate.

To apply CGM-relax one must first select the hyperparameter λ that controls the magnitude of the constraint penalty. In our experiments (see Section 6), we perform a grid search and choose the λ for which the calibration constraints are approximately satisfied and the KL to the base model is small. In CGM-reward, on the other hand, one first pre-computes $\widehat{\boldsymbol{\alpha}}$. While this step also introduces a hyperparameter, the number of Monte Carlo samples N, we find in our experiments that CGM-reward is not sensitive to this choice.

We implement CGM-relax and CGM-reward in our codebase at `https://github.com/smithhenryd/cgm`.

4. Calibrating Neural-SDEs

We next specialize CGM-relax and CGM-reward to neural stochastic differential equations (SDEs). Neural-SDEs are a class of generative models that includes diffusion generative models (Ho et al., 2020; Song et al., 2020; 2021) and have been widely adopted for *de novo* protein design (Watson et al., 2023), medical image reconstruction (Song et al., 2022), image generation (Dhariwal & Nichol, 2021) among other applications. We first introduce neural-SDEs. We then characterize conditions under which the solutions to the calibration and maximum entropy problems coincide for neural-SDEs. Lastly, we describe computational details necessary to implement CGM-relax and CGM-reward.

Background on neural-SDEs. A d-dimensional neural stochastic differential equation is a generative model $p_\theta(\boldsymbol{x})$ defined over continuous paths $\boldsymbol{x} = (\boldsymbol{x}(t))_{0\le t\le 1}$ with $\boldsymbol{x}(t) \in \mathbb{R}^d$ governed by an SDE as

$$p_\theta(\boldsymbol{x}) : d\boldsymbol{x}(t) = b_\theta(\boldsymbol{x}(t), t)dt + \sigma(t)d\boldsymbol{w}(t),\ \boldsymbol{x}(0) \sim p_{\text{init}}.$$

p_{init} is an initial distribution from which sampling is tractable and for $0 \le t \le 1$, the drift $b_\theta : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ is parameterized by a neural-network, $\sigma : [0,1] \to \mathbb{R}_+$ is a diffusion coefficient, and $(\boldsymbol{w}(t))_{0\le t\le 1}$ is a d-dimensional Brownian motion. In the case of a diffusion model $p_{\theta_{\text{base}}}$ is trained by score matching (Song et al., 2020).

Relationship between calibration and maximum entropy solutions. Recall that the calibration and maximum entropy problems, equations (1) and (5), respectively, are related insofar as their optima, p_{θ^*} and $p_{\boldsymbol{\alpha}^*}$, respectively, will be close when the class of generative models is sufficiently expressive. For the case of a neural-SDE base model, we identify sufficient conditions under which these solutions coincide *exactly*.

Theorem 4.1. *Suppose $p_{\theta_{base}}$ is a neural-SDE with drift $b_{\theta_{base}}$. If $\boldsymbol{h}$ is a bounded, continuous function of $\boldsymbol{x}(1)$, and if $\boldsymbol{x}(0)$ and $\boldsymbol{x}(1)$ are independent under $p_{\theta_{base}}$, then the solution to the maximum entropy problem is a neural-SDE with drift*

$$b_{\boldsymbol{\alpha}^*}(\boldsymbol{x}, t) = b_{\theta_{\text{base}}}(\boldsymbol{x}, t) + \sigma(t)^2 \nabla_{\boldsymbol{x}} \log \mathbb{E}_{p_{\theta_{\text{base}}}}\left[\exp\{r_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))\} \mid \boldsymbol{x}(t) = \boldsymbol{x}\right]. \tag{12}$$

The significance of Theorem 4.1 is that, with an expressive enough parametric class of neural network drift functions b_θ, the optimal drift in equation (12) may be approximated to arbitrary precision, and so p_{θ^*} can be made arbitrarily close to $p_{\boldsymbol{\alpha}^*}$. As a consequence, the global minimizer of $\mathcal{L}^{\text{reward}}$ will be very close to satisfying the calibration constraint.

A key condition of Theorem 4.1 is the independence of $\boldsymbol{x}(0)$ and $\boldsymbol{x}(1)$ under $p_{\theta_{\text{base}}}$. In the context of reward fine-tuning, Domingo-Enrich et al. (2025) call this assumption "memorylessness" and find that when it is violated the tilted target $p_{\boldsymbol{\alpha}^*}(\boldsymbol{x})$ cannot be represented by a neural-SDE since then $p_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(0)) \neq p_{\text{init}}(\boldsymbol{x}(0))$. However, Denker et al. (2024) demonstrate a setting where exact memorylessness does not hold, but there exists a neural-SDE that is close in total variation distance to $p_{\boldsymbol{\alpha}^*}$.

CGM-relax and CGM-reward for neural-SDEs. Implementing CGM-relax and CGM-reward requires the ability to draw independent samples from p_θ, to compute density ratios $p_\theta(\boldsymbol{x})/p_{\theta'}(\boldsymbol{x})$, and to take gradients of these density ratios with respect to θ. Each of these steps is possible with minimal approximation error when p_θ is a neural-SDE.

To draw samples from a neural-SDE, one simulates paths using a numerical SDE solver. For our experiments, we will use Euler-Maruyama, which is widely used for diffusion models (Song et al., 2020) and incurs approximation error proportional to the square root of the size of the discretization grid (Karatzas & Shreve, 2012; Oksendal, 2013).

Computation of density ratios requires further notational precision. So far, we have been imprecise insofar as $p_{\theta_{\text{base}}}(\boldsymbol{x})$ and $p_\theta(\boldsymbol{x})$ represent probability measures on the space of d-dimensional continuous paths, rather than probability densities with respect to a common measure on Euclidean space. Nonetheless, Girsanov's Theorem (Appendix C, Theorem C.1) characterizes this density ratio as an integral over $t \in [0,1]$ that can be computed numerically with minimal discretization error. This result allows computation of the terms in the KL divergence estimates and weights in Algorithms 1 and 2. We provide further details in Appendix C.

5. Related Work

Calibration problem. Prior works have addressed problems related to the calibration problem. Wu et al. (2020) propose a training objective for generative adversarial networks

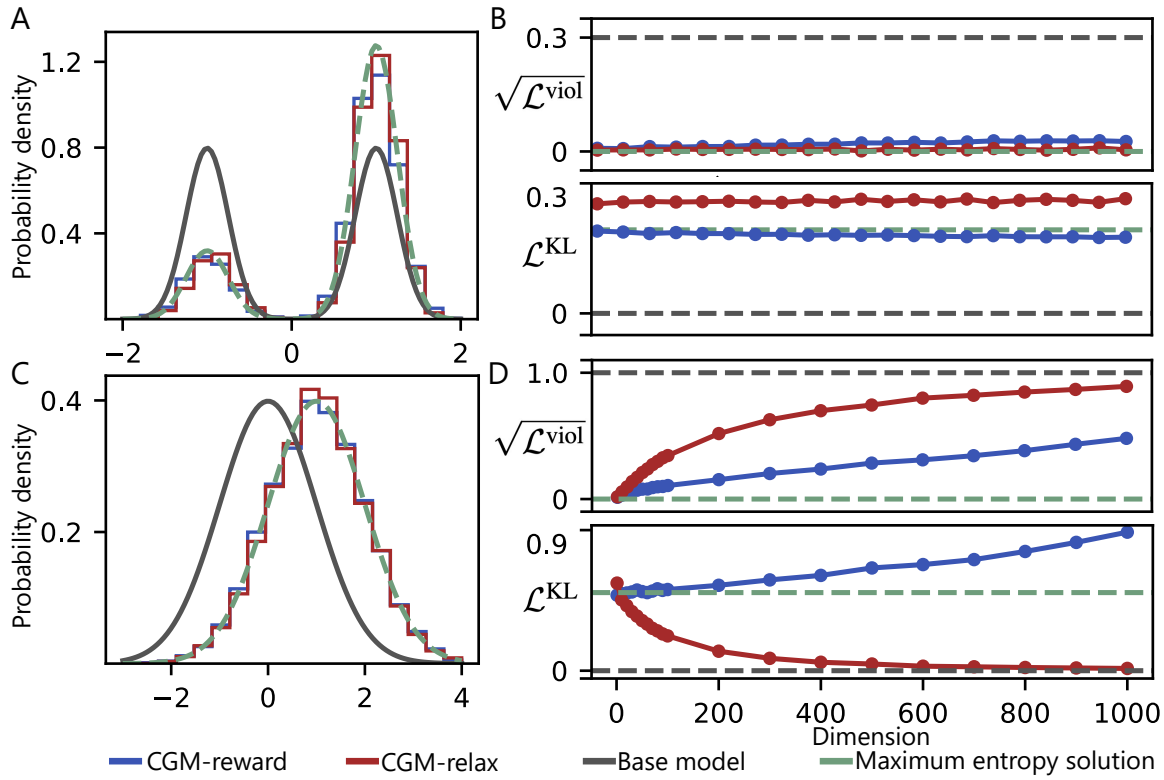

Figure 1. **A & B**: Calibrating mixture proportions in a Gaussian mixture model (GMM). Both CGM-relax and CGM-reward recover the analytical solution to the calibration problem in 1D (A). Even in high-dimensional GMMs, both methods approximately satisfy the desired constraint and achieve KL divergence to the base model that is near optimal (B). KL divergence is computed in nats. **C & D**: Calibrating the mean of a d-dimensional Gaussian distribution. Similar to A, both methods recover the known solution in 1D (C). As the number of calibration constraints grows with the dimension, CGM-reward more faithfully solves the calibration problem (D).

that includes a penalty term similar to $\widehat{\mathcal{L}}^{\text{viol}}$ that encourages model agreement with statistics of the training data. Ganchev et al. (2010) build on Theorem 3.1 to impose constraints on the Bayesian posterior of a probabilistic model. In the context of molecular dynamics simulations, Różycki et al. (2011); Köfinger et al. (2019); Bottaro et al. (2020) leverage Theorem 3.1 to reweight Monte Carlo samples of molecular configurations to match experimental observations of ensemble averages.

Reward fine-tuning for generative models. Several recent works address fine-tuning of neural-SDE models to maximize a pre-specified reward function. For discrete-time denoising diffusion models (Ho et al., 2020), Fan & Lee (2023); Fan et al. (2023); Wallace et al. (2024); Black et al. (2024) frame reward fine-tuning as a reinforcement learning problem. Xu et al. (2023); Clark et al. (2024) instead evaluate the reward on predictions of the denoised state made at various time points and directly perform backpropagation. Tang (2024); Uehara et al. (2024); Domingo-Enrich et al. (2025) extend the reward fine-tuning framework to neural-SDE models and demonstrate that maximizing the expected reward with a KL penalization term is equivalent to solving a stochastic optimal control (SOC) problem. As we pointed out, the solution to the reward fine-tuning problem with choice of reward $r_{\widehat{\alpha}}$ coincides with optimizing the reward loss, and so the algorithms in these works may be applicable to minimizing $\mathcal{L}^{\text{reward}}$.

Conditional generation. Our method is related to conditional sampling methods for neural-SDE models, most of which aim to learn an approximation to a time-dependent likelihood term (Dhariwal & Nichol, 2021; Song et al., 2022; Jalal et al., 2021; Rout et al., 2023; Song et al., 2023; Denker et al., 2024). In particular, Denker et al. (2024) establish a connection between conditional generation and reward fine-tuning. Choosing the calibration constraint $\boldsymbol{h}$ to be the indicator function of an event A and taking the limit $\boldsymbol{h}^* \to 1$ is analogous to the task of conditional generation; in this limit, equation (12) becomes $b_{\theta_{\text{base}}}(\boldsymbol{x}, t) + \sigma(t)^2 \nabla_{\boldsymbol{x}} \log \mathbb{P}_{p_{\theta_{\text{base}}}}(\boldsymbol{x}(1) \in A \mid \boldsymbol{x}(t) = \boldsymbol{x})$, which is the "classifier guidance" term from Dhariwal & Nichol (2021).

6. Experiments

In this section, we evaluate the capability of CGM-relax and CGM-reward to calibrate neural-SDEs. First, we evaluate the performance of our algorithms on two toy examples.

These experiments provide insights into when we would expect each algorithm to approximately solve the calibration problem (1). We then apply CGM-relax to calibrate the Genie2 protein backbone diffusion model (Lin et al., 2024) to match the secondary structure composition of naturally occurring proteins. Appendix D provides additional information regarding our experimental setup.

6.1. Synthetic Data Experiments

We first evaluate CGM-relax and CGM-reward on two synthetic calibration problems. We start by describing each of these two problems, which we select because the maximum entropy problem (5) admits a closed-form solution. We find that in one dimension, CGM-relax and CGM-reward recover the maximum entropy solutions to high accuracy. We further examine the dependence of each algorithm on (i) the dimension of the problem (i.e. $\boldsymbol{x}$) and (ii) the dimension of the constraint $\boldsymbol{h}$. We find that both algorithms can perform well, even in settings where problem dimension is large and the constraint dimension is small as well as in settings where both the problem and constraint dimension are moderately large.

Experimental setup. Across all of our synthetic data experiments, we model the base neural-SDE $p_{\theta_{\text{base}}}$ as the true reversal of the forward noising process. This choice allows us to isolate the effect of calibration (as opposed to training $p_{\theta_{\text{base}}}$ via score matching). We parameterize the calibrated model p_θ as having drift equal to that of $p_{\theta_{\text{base}}}$ plus a neural network offset. The output layer of the neural network is initialized to zero so that the calibrated and pre-trained models are equal at initialization. For CGM-relax, we perform a grid search over possible values of λ for one-dimensional $\boldsymbol{x}$. For CGM-reward, we use $M = 10^4$ samples to compute $\widehat{\boldsymbol{\alpha}}$. All results reported are averages over 10 replications.

For our first synthetic data problem, we calibrate mixture proportions in a Gaussian mixture model (GMM). In particular, our base model $p_{\theta_{\text{base}}}$ generates samples from a GMM with modes at $\boldsymbol{\mu}^{(1)}, \boldsymbol{\mu}^{(2)} \in \mathbb{R}^d$, isotropic noise with variance $\sigma^2 = 0.25$, and mixture proportions $\pi^{(1)} = 1 - \pi^{(2)} = 0.5$. We choose the means such that all the coordinates are zero except $-(\boldsymbol{\mu}^{(1)})[1] = (\boldsymbol{\mu}^{(2)})[1] = 1$, where $\boldsymbol{v}[i]$ denotes the ith component of a vector $\boldsymbol{v}$. In other words, the means of the mixture components differ only along the first dimension (Figure 1A). We then calibrate the pre-trained neural-SDE using a single constraint $\boldsymbol{h}(\boldsymbol{x}) = \mathbb{1}\{\boldsymbol{x}(1)[1] \geq 0\}$, where $\mathbb{1}\{A\}$ is the indicator function of event A and $\boldsymbol{h}^* = 0.8$. Since the components of the mixture model are well-separated, $\boldsymbol{h}$ intuitively captures the proportion of probability mass that lies in the second mixture component. Indeed, in this setting there exists a closed-form solution to the maximum entropy problem that resembles (but is not exactly equal to) another mixture model with mixing proportions $1 - \boldsymbol{h}^*, \boldsymbol{h}^*$ (Figure 1A). We perform CGM-relax with $\lambda = 2 \cdot 10^2$.

Second, we calibrate the mean of a d-dimensional standard multivariate normal distribution (Figure 1C). Observe that in order to specify the mean of a d-dimensional distribution, we need d calibration constraints $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{x}(1)$. To ensure that the KL distance between the base model and the calibration solution (1) remains constant in dimension, we fix $\boldsymbol{h}^*$ to be the all-ones vector, scaled by $d^{-1/2}$. It is straightforward to verify that the solution to the maximum entropy problem (5) is another multivariate Gaussian distribution with mean $\boldsymbol{h}^*$ and identity covariance (Figure 1C). We perform CGM-relax with $\lambda = 10^2$ scaled by d^{-1}, the number of constraints.

Recovery of maximum entropy solution. For both the GMM mixture proportions and Gaussian mean calibration problems, we observe that in one dimension CGM-relax and CGM-reward correctly calibrate the base neural-SDE to the known maximum entropy solution (Figure 1A,C). This is supported by Figure 1B,D, which demonstrate that for low-dimensional problems, both CGM-relax and CGM-reward attain KL to the base model, $\mathcal{L}^{\text{KL}}$, that is near optimal (dashed green line) and small violation of the calibration constraint $\mathcal{L}^{\text{viol}}$.

Scaling in dimension. Of particular relevance to modern applications of neural-SDEs is the question of how our algorithms perform as the dimension of the problem i.e. $\boldsymbol{x}$ and of the constraint $\boldsymbol{h}$ grows large. We observe for the one-dimensional mixture proportion constraint, both CGM-relax and CGM-reward achieve $\mathcal{L}^{\text{KL}}$ that is close to that of the maximum entropy solution as well as small constraint violation $\mathcal{L}^{\text{viol}}$ (Figure 1B), even in up to $d = 10^3$ dimensions. Whereas CGM-relax more faithfully represents the calibration constraint, CGM-reward remains closer to the base model. In the d-dimensional Gaussian mean experiments, though, CGM-reward strictly outperforms CGM-relax in moderate and high-dimensional settings (Figure 1D). Even up to $d = 10^2$ moment constraints, CGM-reward accurately calibrates the base neural-SDE to the target moment.

6.2. Calibrating a Protein Design Generative Model

Diffusion generative models have become a central tool in protein design (Trippe et al., 2023; Watson et al., 2023). However, heuristics such as reduced noise during sampling have been necessary to reliably obtain biophysically plausible samples (see, e.g., Yim et al., 2023). Such heuristics introduce deviations from the statistics of natural proteins (Lu et al., 2025) and lead to substantial losses in diversity as compared to structures in the CATH (Sillitoe et al., 2021) dataset. This relationship can pose a trade-off in application between reliable generation of biophysically plausible samples and diversity. We investigate whether this trade-off can be mitigated by calibrating a protein design model to match the statistics of natural proteins.

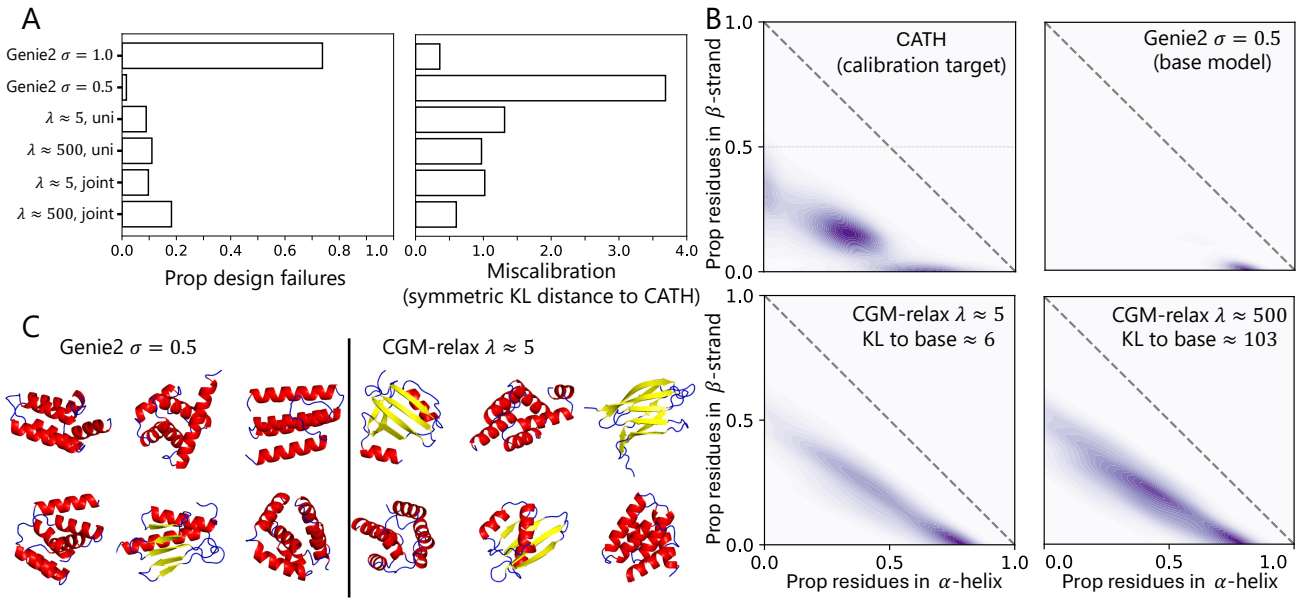

Figure 2. CGM-relax calibrates the secondary structure composition of Genie2 samples to more closely align with CATH domains. **A**: Proportion of design failures, defined as structures having self-consistency RMSD > 2Å, generated by the CGM-relax and Genie2 models as well as the symmetric KL distance of each model's distribution over secondary structure to that of the CATH domains. **B**: Contour plots of the joint distribution over secondary structure. **C**: Example generations from Genie2 and CGM-relax. All plots performing calibration with CGM-relax use $N = 9$ constraints; panels B & C are generated using univariate quantile constraints.

In this section we first provide background on protein design models and their miscalibration. We then define a set of desired calibration constraints and details of our application of CGM fine-tuning on the Genie2 protein backbone model (Lin et al., 2024). Finally, we provide results (summarized in Figure 2) demonstrating successful calibration while maintaining biophysical plausibility.

Proteins, secondary structure, and miscalibration of protein design models. Proteins are a class of molecules that are central to biology and medicine. A key feature of the 3-dimensional structure of proteins is their "backbone", a chain of atoms representable as a list of spatial coordinates. These coordinates adopt canonical geometries relative to one another that may be grouped into three categories ("helix", "strand", and "loop") that define a protein's secondary structure. The secondary structure composition of a protein is the fraction of its backbone atoms in each category.

Although protein design diffusion generative models are typically fit to examples of proteins from nature, Lu et al. (2025) observe that the secondary structure composition of samples from these models deviate systematically from those of natural proteins represented in the CATH database (Sillitoe et al., 2021). In particular, samples are less diverse and under-sample strand secondary structures. This pathology is particularly pronounced for samples generated with heuristics such as reduced noise scaling (Lu et al., 2025).

The top two panels of Figure 2B illustrate the mismatch in secondary structure composition for samples from Genie2 Lin et al. (2024). These samples are generated with a noise-scaling of $\sigma = 0.5$, similar to the recommendations of Lin et al. (2024). Generations without the reduced noise-scaling heuristic agree more closely with CATH statistics (Figure 2A, right), but most appear biophysically implausible as quantified by in silico "self-consistency" or "designability" (see, e.g., Trippe et al., 2023; Watson et al., 2023) (details provided in Appendix D.2).

Calibration constraints. We consider calibration based on the univariate and bivariate cumulative distribution function (CDF) of helical and strand composition. In detail, let $y(\boldsymbol{x}) := [y_\alpha(\boldsymbol{x}), y_\beta(\boldsymbol{x})]$ denote the helical and strand fractions, respectively, of a protein structure. For Genie2 samples, which consist only of C_α carbon backbone coordinates, we compute $y(\boldsymbol{x})$ using the Biotite package (Kunzmann & Hamacher, 2018). For $n = 1, \ldots, N$, let $h_{\alpha,n}(\boldsymbol{x}) = \mathbb{1}\{y_\alpha(\boldsymbol{x}) \leq \alpha_n\}$ and $h_{\beta,n}(\boldsymbol{x}) = \mathbb{1}\{y_\beta(\boldsymbol{x}) \leq \beta_n\}$ denote indicators of whether these proportions are less than or equal to the n/N quantile of $y(\boldsymbol{x})$ across CATH proteins. Finally, let $\boldsymbol{h}(\boldsymbol{x}) = [h_{\alpha,1}h_{\beta,1}, h_{\alpha,1}h_{\beta,2}, \ldots, h_{\alpha,N}h_{\beta,N-1}]$ and $\boldsymbol{h}^*$ be the empirical mean of $\boldsymbol{h}$ across CATH domains. The above defines an (N^2-1)-dimensional joint calibration constraint. We also consider univariate quantile constraints, in which case $\boldsymbol{h}(\boldsymbol{x}) = [h_{\alpha,1}, \ldots, h_{\alpha,N-1}, h_{\beta,1}, \ldots, h_{\beta,N-1}]$ defines a $2(N-1)$-dimensional calibration constraint.

Calibration details. Genie2 (Lin et al., 2024) is a $\sim$15M-parameter SE(3)-equivariant diffusion generative model of protein backbones. Although Genie2 is trained as a discrete-

time diffusion model (see, e.g., Ho et al., 2020), in our experiments we first reparameterize it as a neural SDE following (Song et al., 2020, Appendix B) by viewing it as a discretization of a "variance preserving" SDE. This reparameterization allows sampling with 10-fold fewer steps (10^2 vs. 10^3) using a non-uniform time grid. We implement a $\sigma = 0.5$ noise scaling by multiplying the diffusion coefficient by σ. We set the length of generations to be 100 residues (i.e. atoms); with 3 spatial dimensions, each $\boldsymbol{x}(1)$ is 300 dimensional. Just as we did for the Gaussian mean experiments, we divide the λ hyperparameter in CGM-relax by the number of constraints.

In each calibration run, we fine-tune Genie2 for roughly 52 NVIDIA H100 GPU-hours on the Marlowe cluster (Kapfer et al., 2025) with data parallelism across 4 GPUs.

Results. Figure 2 demonstrates that Genie2 calibrated with CGM-relax generates backbones with secondary structure composition similar to CATH proteins while maintaining high designability. The similarity to CATH proteins, quantified by symmetrized KL divergence based on kernel density estimates, is lower than $p_{\theta_{\text{base}}}$ (< 1 nat versus > 3 nats). This similarity is comparable to the base model when sampled without noise-scaling (Figure 2A). However, in contrast to the model without noise-scaling, this high similarity is obtained with high designability; fewer than 20% of all CGM-relax fine-tuned generations are not designable versus 75% for the base model without noise-scaling. Figure 2C illustrates sample generations from the base Genie2 and CGM-relax calibrated models; Appendix Figure 4 provides additional samples.

Among the CGM-relax calibrated models, we observe that larger penalties on $\mathcal{L}^{\text{viol}}$ (i.e. λ) result in secondary structure compositions more closely aligned with the CATH domains. However, a smaller fraction of these generations are designable (Figure 2A). Models calibrated with joint CDF constraints ('joint') generate structures more similar in secondary structure content to the CATH domains compared to those calibrated with univariate constraints ('uni') but produce fewer designable structures.

An initial exploration of CGM-reward proved less fruitful. We found that CGM-reward was less successful at satisfying the calibration constraints. Provided the success of CGM-reward in high-dimensional regimes for our synthetic data experiments, we will investigate its failure in future work.

7. Discussion

Despite a large and growing body of work on algorithms for fine-tuning generative models, algorithms for calibrating these models remain underexplored. Presumably, this is due in part to the relative complexity of the task compared to reward fine-tuning, which maximizes the expected reward plus a KL penalty; the reward is defined at the level of single samples. CGM-relax and CGM-reward provide first steps by framing calibration as a constrained optimization problem and searching for approximate solutions.

However, our work has limitations and presents several directions for future investigation. A first limitation is that each of CGM-relax and CGM-reward aims to solve a high-dimensional, non-convex optimization problem via stochastic gradients. The fundamental difficulty of non-convex optimization is likely responsible for much of the deviation of our calibrated models from the solution to the calibration problem (1). Second, we have yet to investigate how the capabilities of CGM-relax and CGM-reward depend on the extent of miscalibration of the base model, and what algorithmic variations or alternatives might be required for calibration constraints that are more difficult to satisfy. A third limitation is that it is not clear how to calibrate to infinite-dimensional constraints, for example, to a full cumulative distribution function of a statistic rather than its quantiles. These questions remain open for future work.

Acknowledgments

We thank Julius Berner for his numerous helpful discussions about the connection between the calibration problem and the reward fine-tuning problem for neural-SDE models as well as the inclusion of a baseline in the CGM-relax and CGM-reward gradient estimates. We thank Tianyu Lu for providing us with the dataset of secondary structure composition for CATH domains. This dataset appeared in Lu et al. (2025). We thank Zhaoyang Li for his assistance with the self-consistency evaluations. We thank Zoe Ryan for her help parallelizing CGM-relax for Genie2 across GPUs.

Henry Smith is supported by the NSF Graduate Research Fellowship (DGE-2146755) and the Knight-Hennessy Graduate Fellowship.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Black, K., Janner, M., Du, Y., Kostrikov, I., and Levine, S. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, 2024.

Bottaro, S., Bengtsen, T., and Lindorff-Larsen, K. Integrating molecular simulation and experimental data: a

Bayesian/maximum entropy reweighting approach. *Structural Bioinformatics: Methods and Protocols*, 2020.

Cameron, R. H. and Martin, W. T. Transformations of Wiener integrals under translations. *Annals of Mathematics*, 1944.

Clark, K., Vicol, P., Swersky, K., and Fleet, D. J. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, 2024.

Csiszár, I. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, 1975.

Dauparas, J., Anishchenko, I., Bennett, N., Bai, H., Ragotte, R. J., Milles, L. F., Wicky, B. I. M., Courbet, A., de Haas, R. J., Bethel, N., Leung, P. J. Y., Huddy, T. F., Pellock, S., Tischer, D., Chan, F., Koepnick, B., Nguyen, H., Kang, A., Sankaran, B., Bera, A. K., King, N. P., and Baker, D. Robust deep learning–based protein sequence design using ProteinMPNN. *Science*, 2022.

Denker, A., Vargas, F., Padhy, S., Didi, K., Mathis, S., Barbano, R., Dutordoir, V., Mathieu, E., Komorowska, U. J., and Lio, P. DEFT: efficient fine-tuning of diffusion models by learning the generalised h-transform. *Advances in Neural Information Processing Systems*, 2024.

Dhariwal, P. and Nichol, A. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, 2021.

Domingo-Enrich, C., Drozdzal, M., Karrer, B., and Chen, R. T. Adjoint matching: fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *International Conference on Learning Representations*, 2025.

Fan, Y. and Lee, K. Optimizing DDPM sampling with shortcut fine-tuning. In *International Conference on Machine Learning*, 2023.

Fan, Y., Watkins, O., Du, Y., Liu, H., Ryu, M., Boutilier, C., Abbeel, P., Ghavamzadeh, M., Lee, K., and Lee, K. DPOK: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 2023.

Friedman, A. Stochastic differential equations and applications. In *Stochastic Differential Equations*. Springer, 1975.

Ganchev, K., Graça, J., Gillenwater, J., and Taskar, B. Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research*, 2010.

Girsanov, I. V. On transforming a certain class of stochastic processes by absolutely continuous substitution of measures. *Theory of Probability & Its Applications*, 1960.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 2020.

Jalal, A., Arvinte, M., Daras, G., Price, E., Dimakis, A. G., and Tamir, J. Robust compressed sensing MRI with deep generative priors. *Advances in Neural Information Processing Systems*, 2021.

Kapfer, C., Stine, K., Narasimhan, B., Mentzel, C., and Candès, E. Marlowe: Stanford's GPU-based computational instrument, 2025.

Karatzas, I. and Shreve, S. *Brownian Motion and Stochastic Calculus*, volume 113. Springer Science & Business Media, 2012.

Kobak, D., González-Márquez, R., Horvát, E.-Á., and Lause, J. Delving into ChatGPT usage in academic writing through excess vocabulary. *arXiv preprint arXiv:2406.07016*, 2024.

Köfinger, J., Stelzl, L. S., Reuter, K., Allande, C., Reichel, K., and Hummer, G. Efficient ensemble refinement by reweighting. *Journal of Chemical Theory and Computation*, 2019.

Kool, W., van Hoof, H., and Welling, M. Buy 4 REINFORCE samples, get a baseline for free! In *"Deep* RL *Meets Structured Prediction" Workshop at the International Conference on Learning Representations*, 2019.

Kullback, S. *Information Theory and Statistics*. John Wiley & Sons, 1959.

Kullback, S. and Khairat, M. A note on minimum discrimination information. *The Annals of Mathematical Statistics*, 1966.

Kunzmann, P. and Hamacher, K. Biotite: a unifying open source computational biology framework in Python. *BMC Bioinformatics*, 2018.

Lavenberg, S. S. and Welch, P. D. A perspective on the use of control variables to increase the efficiency of Monte Carlo simulations. *Management Science*, 1981.

Léonard, C. Some properties of path measures. *Séminaire de Probabilités*, 2014.

Lewis, S., Hempel, T., Jiménez-Luna, J., Gastegger, M., Xie, Y., Foong, A. Y., Satorras, V. G., Abdin, O., Veeling, B. S., Zaporozhets, I., Chen, Y., Yang, S., Schneuing, A., Nigam, J., Barbero, F., Stimper, V., Campbell, A., Yim, J., Lienen, M., Shi, Y., Zheng, S., Schulz, H., Munir, U.,

Clementi, C., and Noé, F. Scalable emulation of protein equilibrium ensembles with generative deep learning. *bioRxiv*, 2024.

Lin, Y. and Nguyen, H. C. In-silico Protein Design Pipeline, 2024.

Lin, Y., Lee, M., Zhang, Z., and AlQuraishi, M. Out of many, one: Designing and scaffolding proteins at the scale of the structural universe with Genie 2. *arXiv preprint arXiv:2405.15489*, 2024.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., dos Santos Costa, A., Fazel-Zarandi, M., Sercu, T., Candido, S., and Rives, A. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 2023.

Lombardi, L. E., Martí, M. A., and Capece, L. CG2AA: backmapping protein coarse-grained structures. *Bioinformatics*, 2016.

Lu, T., Liu, M., Chen, Y., Kim, J., and Huang, P.-S. Assessing generative model coverage of protein structures with SHAPES. *bioRxiv*, 2025.

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. Monte Carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 2020.

Nüsken, N. and Richter, L. Solving high-dimensional Hamilton–Jacobi–Bellman PDEs using neural networks: perspectives from the theory of controlled diffusions and measures on path space. *Partial Differential Equations and Applications*, 2021.

Oksendal, B. *Stochastic Differential Equations: An Introduction with Applications*. Springer Science & Business Media, 2013.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 2023.

Ranganath, R., Gerrish, S., and Blei, D. Black box variational inference. In *Artificial Intelligence and Statistics*. PMLR, 2014.

Richter, L., Boustati, A., Nüsken, N., Ruiz, F., and Akyildiz, O. D. Vargrad: a low-variance gradient estimator for variational inference. *Advances in Neural Information Processing Systems*, 2020.

Rockafellar, R. T. *Convex Analysis*. Princeton University Press, 1997.

Rout, L., Raoof, N., Daras, G., Caramanis, C., Dimakis, A., and Shakkottai, S. Solving linear inverse problems provably via posterior sampling with latent diffusion models. *Advances in Neural Information Processing Systems*, 2023.

Różycki, B., Kim, Y. C., and Hummer, G. SAXS ensemble refinement of ESCRT-III CHMP3 conformational transitions. *Structure*, 2011.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Sillitoe, I., Bordin, N., Dawson, N., Waman, V. P., Ashford, P., Scholes, H. M., Pang, C. S. M., Woodridge, L., Rauer, C., Sen, N., Abbasian, M., Le Cornu, S., Lam, S. D., Berka, K., Hutařová Vareková, I., Svobodova, R., Lees, J., and Orengo, C. A. CATH: increased structural coverage of functional space. *Nucleic Acids Research*, 2021.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Song, J., Vahdat, A., Mardani, M., and Kautz, J. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2023.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Song, Y., Shen, L., Xing, L., and Ermon, S. Solving inverse problems in medical imaging with score-based generative models. In *International Conference on Learning Representations*, 2022.

Tang, W. Fine-tuning of diffusion models via stochastic control: entropy regularization and beyond. *arXiv preprint arXiv:2403.06279*, 2024.

Tohme, T., Sadr, M., Youcef-Toumi, K., and Hadjiconstantinou, N. G. MESSY estimation: Maximum-entropy based stochastic and symbolic density estimation. *arXiv preprint arXiv:2306.04120*, 2023.

Trippe, B. L., Yim, J., Tischer, D., Baker, D., Broderick, T., Barzilay, R., and Jaakkola, T. Diffusion probabilistic modeling of protein backbones in 3D for the motif-scaffolding problem. In *International Conference on Learning Representations*, 2023.

Uehara, M., Zhao, Y., Black, K., Hajiramezanali, E., Scalia, G., Diamant, N. L., Tseng, A. M., Biancalani, T., and Levine, S. Fine-tuning of continuous-time diffusion

models as entropy-regularized control. *arXiv preprint arXiv:2402.15194*, 2024.

Wainwright, M. J., Jordan, M. I., et al. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 2008.

Wallace, B., Dang, M., Rafailov, R., Zhou, L., Lou, A., Purushwalkam, S., Ermon, S., Xiong, C., Joty, S., and Naik, N. Diffusion model alignment using direct preference optimization. In *Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2024.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., Wicky, B. I. M., Hanikel, N., Pellock, S. J., Courbet, A., Sheffler, W., Wang, J., Venkatesh, P., Sappington, I., Vázquez Torres, S., Lauko, A., De Bortoli, V., Mathieu, E., Ovchinnikov, S., Barzilay, R., Jaakkola, T. S., DiMaio, F., Baek, M., and Baker, D. De novo design of protein structure and function with RFdiffusion. *Nature*, 2023.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 1992.

Wu, J.-L., Kashinath, K., Albert, A., Chirila, D., Xiao, H., et al. Enforcing statistical constraints in generative adversarial networks for modeling chaotic dynamical systems. *Journal of Computational Physics*, 2020.

Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., and Dong, Y. ImageReward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 2023.

Yim, J., Trippe, B. L., De Bortoli, V., Mathieu, E., Doucet, A., Barzilay, R., and Jaakkola, T. SE(3) diffusion model with application to protein backbone generation. In *International Conference on Machine Learning*, 2023.

Appendix Contents

A. CGM-relax and CGM-reward Algorithms

In this section, we provide specifics of the CGM-relax and CGM-reward algorithms. In particular, we will prove that the estimates for the relaxed and reward losses, $\widehat{\mathcal{L}}^{\text{relax}}$ and $\widehat{\mathcal{L}}^{\text{reward}}$ (Section 3.1 and 3.2), are unbiased. We will also discuss in greater detail how to compute the unbiased gradient estimators for the losses introduced in Section 3.3.

Throughout this section we will make the following regularity assumptions on the calibrated model p_θ and the constraint functions $\boldsymbol{h}$. First, we assume that $\nabla_{\tilde{\theta}} p_{\tilde{\theta}}(\boldsymbol{x})/p_\theta(\boldsymbol{x}), \log p_{\tilde{\theta}}(\boldsymbol{x}), \nabla_{\tilde{\theta}} \log p_{\tilde{\theta}}(\boldsymbol{x})$ are uniformly dominated by a function that is square integrable with respect to $p_\theta(\boldsymbol{x})$, for all $\tilde{\theta}$ belonging to some neighborhood of θ. Second, we assume that $\boldsymbol{h}(\boldsymbol{x})$ has finite fourth moment under $p_\theta(\boldsymbol{x})$. These assumptions are necessary to exchange integration and differentiation in Appendix A.2 via Dominated Convergence. For neural-SDEs, for instance, these assumptions are satisfied when the controller u_θ and its gradient ∇u_θ satisfy a linear growth condition (see Appendix C).

A.1. Loss Estimates

We begin by proving that our estimates for $\mathcal{L}^{\text{relax}}$ and $\mathcal{L}^{\text{reward}}$ are, on average, correct. We acknowledge that unbiasedness is not the end all be all. For instance, for the term $\mathcal{L}^{\text{viol}} = \|\mathbb{E}_{p_\theta}[\boldsymbol{h}(\boldsymbol{x})] - \boldsymbol{h}^*\|^2$ that appears the relaxed loss, there exists a biased estimator, namely $\max\{\widehat{\mathcal{L}}^{\text{viol}}, 0\}$, that strictly dominates our estimator $\widehat{\mathcal{L}}^{\text{viol}}$ in terms of risk (in particular, its variance is smaller). Nonetheless, the notion of unbiasedness is a useful one.

Proposition A.1. *$\widehat{\mathcal{L}}^{\text{relax}}$ is unbiased for the relaxed loss $\mathcal{L}^{\text{relax}}$.*

Proof. Following our presentation in Section 3.1, we prove unbiasedness of $\widehat{\mathcal{L}}^{\text{relax}}$ by showing that $\widehat{\mathcal{L}}^{\text{KL}}$ is unbiased for $\mathcal{L}^{\text{KL}} = \mathrm{D}_{\text{KL}}\left(p_\theta \parallel p_{\theta_{\text{base}}}\right)$ and that $\widehat{\mathcal{L}}^{\text{viol}}$ is unbiased for $\mathcal{L}^{\text{viol}}$, the squared norm difference between the expectation of $\boldsymbol{h}$ under p_θ and the target expectation $\boldsymbol{h}^*$.

As for $\widehat{\mathcal{L}}^{\text{KL}}$, its expectation is

$$\mathbb{E}_{p_\theta}\left[\widehat{\mathcal{L}}^{\text{KL}}\right] = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{p_\theta}\left[\log \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)}\right] = \frac{1}{M}\sum_{m=1}^{M} \mathrm{D}_{\text{KL}}\left(p_\theta \parallel p_{\theta_{\text{base}}}\right) = \mathrm{D}_{\text{KL}}\left(p_\theta \parallel p_{\theta_{\text{base}}}\right).$$

In the first equality we invoke the linearity of expectation and in the second we use our assumption that $\{\boldsymbol{x}_m\}_{m=1}^M$ are sampled from p_θ. And for $\widehat{\mathcal{L}}^{\text{viol}}$, we first recall the bias-variance decomposition for a real-valued random variable Z and scalar constant c:

$$\mathbb{E}[(Z-c)^2] = (\mathbb{E}[Z]-c)^2 + \text{Var}(Z).$$

By applying this result to each dimension of $\frac{1}{M}\sum_{m=1}^M \tilde{\boldsymbol{h}}_m = \frac{1}{M}\sum_{m=1}^M (\boldsymbol{h}(\boldsymbol{x}_m) - \boldsymbol{h}^*)$ with $c = 0$, we obtain

$$\mathbb{E}_{p_\theta}\left\|\frac{1}{M}\sum_{m=1}^{M}\tilde{\boldsymbol{h}}_m\right\|^2 = \|\mathbb{E}_{p_\theta}[\tilde{\boldsymbol{h}}(\boldsymbol{x})]\|^2 + \frac{1}{M}\mathbb{E}_{p_\theta}\|\tilde{\boldsymbol{h}}(\boldsymbol{x}) - \mathbb{E}_{p_\theta}[\tilde{\boldsymbol{h}}(\boldsymbol{x})]\|^2, \tag{13}$$

where $\tilde{\boldsymbol{h}}(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{h}^*$. Next, we replace the final term $\frac{1}{M}\mathbb{E}_{p_\theta}\|\tilde{\boldsymbol{h}}(\boldsymbol{x}) - \mathbb{E}_{p_\theta}[\tilde{\boldsymbol{h}}(\boldsymbol{x})]\|^2$ in the above expression with $\mathbb{E}_{p_\theta}[\frac{1}{M(M-1)}\sum_m \|\tilde{\boldsymbol{h}}_m - \frac{1}{M}\sum_{m'}\tilde{\boldsymbol{h}}_{m'}\|^2]$. The quantity $\frac{1}{M(M-1)}\sum_m \|\tilde{\boldsymbol{h}}_m - \frac{1}{M}\sum_{m'}\tilde{\boldsymbol{h}}_{m'}\|^2$ is simply the sample variance of $\{\tilde{\boldsymbol{h}}_m\}_{m=1}^M$, summed across each dimension and scaled by $\frac{1}{M}$. The sample variance of $\{\tilde{\boldsymbol{h}}_m\}_{m=1}^M$ is unbiased for $\text{Var}[\tilde{\boldsymbol{h}}]$. Rearranging the above expression yields

$$\|\mathbb{E}_{p_\theta}[\tilde{\boldsymbol{h}}(\boldsymbol{x})]\|^2 = \mathbb{E}_{p_\theta}\left\|\frac{1}{M}\sum_{m=1}^{M}\tilde{\boldsymbol{h}}_m\right\|^2 - \frac{1}{M}\mathbb{E}_{p_\theta}\left[\frac{1}{(M-1)}\sum_{m=1}^{M}\left\|\tilde{\boldsymbol{h}}_m - \frac{1}{M}\sum_{m'=1}^{M}\tilde{\boldsymbol{h}}_{m'}\right\|^2\right].$$

This proves that $\widehat{\mathcal{L}}^{\text{viol}}$ is unbiased for $\|\mathbb{E}_{p_\theta}[\boldsymbol{h}(\boldsymbol{x})] - \boldsymbol{h}^*\|^2$. □

Likewise, we demonstrate that our estimator for the reward loss is unbiased.

Proposition A.2. *$\widehat{\mathcal{L}}^{\text{reward}}$ is unbiased for the reward loss $\mathcal{L}^{\text{reward}}$.*

Proof. In the proof of Proposition A.1 we already demonstrated $\frac{1}{M}\sum_{m=1}^M \log \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)}$ is unbiased for $\mathcal{L}^{\text{KL}}$. By an identical argument, $-\frac{1}{M}\sum_{m=1}^M r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_m)$ is unbiased for $\mathcal{L}^{\text{r}} = \mathbb{E}_{p_\theta}[-r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})]$ (again, it is a Monte Carlo estimate). □

A.2. Unbiased Gradient Estimates

Of greater relevance to the CGM-relax and CGM-reward algorithms is computing unbiased estimates of the gradients of $\widehat{\mathcal{L}}^{\text{relax}}$ and $\widehat{\mathcal{L}}^{\text{reward}}$, rather than of the losses themselves. However, as we detailed in Section 3.3, the naïve idea of taking the unbiased loss estimators $\widehat{\mathcal{L}}^{\text{relax}}$, $\widehat{\mathcal{L}}^{\text{reward}}$ and differentiating them with respect to θ will not yield unbiased gradient estimates. This is because the probability distribution with respect to which the expectation is taken also depends on θ, which needs to be taken into account in the gradient estimate.

For CGM-reward, we propose using the gradient estimator

$$\begin{aligned}\widehat{G}^{\text{reward}} &= \frac{1}{M}\sum_{m=1}^{M}(\nabla_\theta \log p_\theta(\boldsymbol{x}_m))\left(\log\frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_m) - \text{LOO}_m\right)\\ \text{LOO}_m &= \frac{1}{M-1}\sum_{m'\neq m}\left(\log\frac{p_\theta(\boldsymbol{x}_{m'})}{p_{\theta_{\text{base}}}(\boldsymbol{x}_{m'})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_{m'})\right),\end{aligned} \tag{14}$$

where LOO_m is the leave-one-out baseline (Kool et al., 2019) corresponding to sample m. The first term in the sum (14) is known as the score-based gradient estimate or, in the terminology of reinforcement learning, the REINFORCE gradient estimate (Williams, 1992). This is because the term $\nabla_\theta \log p_\theta(\boldsymbol{x}_m)$ is known as the "score" of the calibrated model at $\boldsymbol{x}_m$. In Section 3.3 we derived the score-based gradient estimate by rewriting $\mathcal{L}^{\text{reward}}$ as an expectation with respect to another distribution $p_{\theta'}$, differentiating the Monte Carlo estimate to the expectation under $p_{\theta'}$, and evaluating the gradient at the choice $\theta' = \theta$. We formalize this argument in Proposition A.3.

Observe that by independence of the samples $\{\boldsymbol{x}_m\}_{m=1}^M$, it holds that for each $m \neq m'$,

$$\mathbb{E}_{p_\theta}\left[(\nabla_\theta \log p_\theta(\boldsymbol{x}_m))\left\{\log\frac{p_\theta(\boldsymbol{x}_{m'})}{p_{\theta_{\text{base}}}(\boldsymbol{x}_{m'})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_{m'})\right\}\right] = \mathbb{E}_{p_\theta}\left[\nabla_\theta \log p_\theta(\boldsymbol{x}_m)\right]\mathbb{E}_{p_\theta}\left[\log\frac{p_\theta(\boldsymbol{x}_{m'})}{p_{\theta_{\text{base}}}(\boldsymbol{x}_{m'})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_{m'})\right] = 0.$$

Consequently, when proving that $\widehat{G}^{\text{reward}}$ is unbiased for $\nabla_\theta\mathcal{L}^{\text{reward}}$, we can safely ignore the leave-one-out averages. As we suggested in Section 3.3, we observe that the inclusion of the baseline is important for reducing the variance of our gradient estimate.

Proposition A.3. *$\widehat{G}^{\text{reward}}$ is unbiased for the gradient of the reward loss, $\nabla_\theta\mathcal{L}^{\text{reward}}$.*

Proof. We start by writing out the gradient of $\mathcal{L}^{\text{reward}}$ directly:

$$\begin{aligned}\nabla_\theta\mathcal{L}^{\text{reward}}(\theta) &= \nabla_\theta\mathbb{E}_{p_\theta}\left[\log\frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right]\\ &= \nabla_\theta\int p_\theta(\boldsymbol{x})\left\{\log\frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right\}d\boldsymbol{x}\\ &\overset{(\star)}{=} \int(\nabla_\theta p_\theta(\boldsymbol{x}))\left\{\log\frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right\}d\boldsymbol{x} + \int p_\theta(\boldsymbol{x})\nabla_\theta\log p_\theta(\boldsymbol{x})d\boldsymbol{x}\\ &= \mathbb{E}_{p_\theta}\left[(\nabla_\theta\log p_\theta(\boldsymbol{x}))\left\{\log\frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right\}\right] + \mathbb{E}_{p_\theta}[\nabla_\theta\log p_\theta(\boldsymbol{x})].\end{aligned}$$

For the equality $(\star)$, exchange of the gradient and derivative is permissible by the conditions we stated at the beginning of the section via Dominated Convergence. The second term is the gradient is the expected score, which is zero. And so the gradient of the reward loss is

$$\nabla_\theta\mathcal{L}^{\text{reward}}(\theta) = \mathbb{E}_{p_\theta}\left[(\nabla_\theta\log p_\theta(\boldsymbol{x}))\left\{\log\frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right\}\right]. \tag{15}$$

Looking at our gradient estimator $\widehat{G}^{\text{reward}}$ in (14) and ignoring the leave-one-out averages, we see that it is exactly the Monte Carlo estimate of the gradient of $\mathcal{L}^{\text{reward}}$(15). □

Deriving an unbiased gradient estimate for the relaxed loss is more challenging, since the loss cannot be simply expressed as the expectation under p_θ of some objective. It is clear that, in the same as we did for the reward loss, one can compute an unbiased estimate for the gradient of $\mathcal{L}^{\text{KL}}$ in the relaxed loss

$$\widehat{G}_{\text{KL}} = \frac{1}{M}\sum_{m=1}^{M}(\nabla_\theta \log p_\theta(\boldsymbol{x}_m))\left(\log\frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta_{\text{base}}}(\boldsymbol{x}_m)} - \text{LOO}_m\right),\ \text{LOO}_m = \frac{1}{M-1}\sum_{m'\neq m}\log\frac{p_\theta(\boldsymbol{x}_{m'})}{p_{\theta_{\text{base}}}(\boldsymbol{x}_{m'})}.$$

And so it only remains to compute an unbiased gradient estimate for $\mathcal{L}^{\text{viol}}$. To do so, we first recall the unbiased estimator $\widehat{\mathcal{L}}^{\text{viol}}$ for $\mathcal{L}^{\text{viol}}$ that we introduced in Section 3.1

$$\widehat{\mathcal{L}}^{\text{viol}}(\{\tilde{\boldsymbol{h}}_m\}_{m=1}^M) := \left\|\frac{1}{M}\sum_{m=1}^{M}\tilde{\boldsymbol{h}}_m\right\|^2 - \frac{1}{M(M-1)}\sum_{m=1}^{M}\left\|\tilde{\boldsymbol{h}}_m - \frac{1}{M}\sum_{m'=1}^{M}\tilde{\boldsymbol{h}}_{m'}\right\|^2.$$

We then replace $\{\tilde{\boldsymbol{h}}_m\}_{m=1}^M$ by $\{w_m\tilde{\boldsymbol{h}}_m\}_{m=1}^M$, where $w_m = \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta'}(\boldsymbol{x}_m)}$ are weights defined by another probability distribution $p_{\theta'}$. In statistics, $\{w_m\}_{m=1}^M$ are referred to as importance sampling weights with proposal distribution $p_{\theta'}$ and target distribution p_θ. To estimate the gradient of $\|\mathbb{E}_{p_\theta}[\boldsymbol{h}] - \boldsymbol{h}^*\|^2 = \|\mathbb{E}_{p_\theta}[\tilde{\boldsymbol{h}}]\|^2$, we propose computing the gradient of $\widehat{\mathcal{L}}^{\text{viol}}(\{w_m\tilde{\boldsymbol{h}}_m\}_{m=1}^M)$ with respect to θ and then evaluating the result at $\theta' = \theta$. This is equivalent to first evaluating $\widehat{\mathcal{L}}^{\text{viol}}(\{w_m\tilde{\boldsymbol{h}}_m\}_{m=1}^M)$ at $\theta' = \texttt{stop-grad}(\theta)$ and then computing the gradient, where $p_{\texttt{stop-grad}(\theta)}$ is equal in distribution to p_θ but does not track gradients. This yields the overall gradient estimator for the relaxed loss

$$\widehat{G}^{\text{relax}} = \widehat{G}_{\text{KL}} + \lambda\nabla_\theta\widehat{\mathcal{L}}_{\text{const}}\left(\left\{\frac{p_\theta(\boldsymbol{x}_m)}{p_{\texttt{stop-grad}(\theta)}(\boldsymbol{x}_m)}\tilde{\boldsymbol{h}}_m\right\}_{m=1}^M\right).$$

The intuition behind this gradient estimate is that, from Proposition A.1, we know $\widehat{\mathcal{L}}^{\text{viol}}(\{\tilde{\boldsymbol{h}}_m\}_{m=1}^M)$ (i.e. without the importance sampling weights) is unbiased for $\|\mathbb{E}_{p_\theta}[\boldsymbol{h}] - \boldsymbol{h}^*\|^2$. The only reason we cannot directly differentiate this estimate is that $\{\boldsymbol{x}_m\}_{m=1}^M$ depend on θ since they are sampled from p_θ. To address this issue, we instead sample $\{\boldsymbol{x}_m\}_{m=1}^M$ from a different probability distribution, $p_{\theta'}$, that does not depend on θ. In our approach, we take $\theta' = \texttt{stop-grad}(\theta)$. To address the fact that we are no longer sampling from p_θ, we multiply each $\tilde{\boldsymbol{h}}_m$ by the respective importance sampling weight $w_m = \frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta'}(\boldsymbol{x}_m)}$. The most important piece of our argument involves showing that, when we sample $\{\boldsymbol{x}_m\}_{m=1}^M$ *i.i.d.* from $p_{\theta'}$ and replace $\{\tilde{\boldsymbol{h}}_m\}_{m=1}^M$ by $\{w_m\tilde{\boldsymbol{h}}_m\}_{m=1}^M$ in the estimator $\widehat{\mathcal{L}}^{\text{viol}}$, it remains unbiased for $\|\mathbb{E}_{p_\theta}[\tilde{\boldsymbol{h}}]\|^2$. Then, since, the samples $\{\boldsymbol{x}_m\}_{m=1}^M$ no longer depend on θ, we can differentiate the estimate $\widehat{\mathcal{L}}^{\text{viol}}$ to obtain an unbiased gradient estimate.

We make this argument mathematically precise in the following proposition:

Proposition A.4. *$\widehat{G}^{\text{relax}}$ is unbiased for the gradient of the relaxed loss, $\nabla_\theta\mathcal{L}^{\text{relax}}$.*

Proof. From Proposition A.1, we know that $\widehat{G}_{\text{KL}}$ is unbiased for $\nabla_\theta\mathcal{L}^{\text{KL}}$, and so it only remains to verify that the second term is unbiased for $\lambda\nabla_\theta\mathcal{L}^{\text{viol}} = \lambda\nabla_\theta\|\mathbb{E}_{p_\theta}[\boldsymbol{h}] - \boldsymbol{h}^*\|^2$. To this end, by repeating the proof of Proposition A.1 (i.e. using the bias-variance decomposition), it is straightforward to show that

$$\mathbb{E}_{p_{\theta'}}\left[\widehat{\mathcal{L}}^{\text{viol}}\left(\left\{\frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta'}(\boldsymbol{x}_m)}\tilde{\boldsymbol{h}}_m\right\}_{m=1}^M\right)\right] = \left\|\mathbb{E}_{p_{\theta'}}\left[\frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta'}(\boldsymbol{x}_m)}\tilde{\boldsymbol{h}}_m\right]\right\|^2 = \|\mathbb{E}_{p_\theta}[\tilde{\boldsymbol{h}}]\|^2.$$

In other words, $\widehat{\mathcal{L}}^{\text{viol}}(\{w_m\tilde{\boldsymbol{h}}_m\}_{m=1}^M)$ is unbiased for $\mathcal{L}^{\text{viol}}$. However, since the samples $\{\boldsymbol{x}_m\}_{m=1}^M$ are drawn from $p_{\theta'}$, a probability distribution that does not depend on θ, then we can exchange the gradient and expectation by appealing to Dominated Convergence under the assumptions stated at the beginning of the section. In particular, we have

$$\begin{aligned}\mathbb{E}_{p_{\theta'}}\left[\nabla_\theta\widehat{\mathcal{L}}^{\text{viol}}\left(\left\{\frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta'}(\boldsymbol{x}_m)}\tilde{\boldsymbol{h}}_m\right\}_{m=1}^M\right)\right] &= \nabla_\theta\mathbb{E}_{p_{\theta'}}\left[\widehat{\mathcal{L}}^{\text{viol}}\left(\left\{\frac{p_\theta(\boldsymbol{x}_m)}{p_{\theta'}(\boldsymbol{x}_m)}\tilde{\boldsymbol{h}}_m\right\}_{m=1}^M\right)\right] \\ &= \nabla_\theta\mathcal{L}^{\text{viol}},\end{aligned}$$

where the final line follows from the unbiasedness of $\widehat{\mathcal{L}}^{\text{viol}}(\{w_m\tilde{\boldsymbol{h}}_m\}_{m=1}^M)$ for $\mathcal{L}^{\text{viol}}$. □

As we discussed, the key insight from the proof of Proposition A.4 is that, by introducing importance weights, we can compute an unbiased estimate to $\|\mathbb{E}_{p_\theta}[\boldsymbol{h}] - \boldsymbol{h}^*\|^2 = \|\mathbb{E}_{p_\theta}[\tilde{\boldsymbol{h}}]\|$ without sampling directly from p_θ. Notice that in our argument, the only step that relied upon the choice of $\theta' = \texttt{stop-grad}(\theta)$ was when appealing to Dominated Convergence to exchange the gradient and the expectation. One could also sample $\{\boldsymbol{x}_m\}_{m=1}^M$ *i.i.d.* from another distribution $p_{\theta'}$.[1]

B. Maximum Entropy Principle

In this section, we provide a precise statement of the maximum entropy principle as it is related to the calibration problem.

We begin by stating a more general form of Theorem 3.1, which will be necessary for our discussion of neural-SDEs:

Theorem B.1 (Kullback (1959); Kullback & Khairat (1966))**.** *Let $X := (X, \mathcal{X})$ be a measurable space and P be a probability measure defined on X. Moreover, let $\mathcal{S}$ be the collection of probability measures on X that have densities with respect to P and satisfy the calibration constraint $\mathbb{E}_Q[\boldsymbol{h}(\boldsymbol{x})] = \boldsymbol{h}^*$, for $Q \in \mathcal{S}$. Here $\boldsymbol{h} : X \to \mathbb{R}^d$ is assumed to be X-measurable.*

Suppose there exists a probability measure $\mathcal{Q}^*$ belonging to $\mathcal{S}$ with density of the form $\mathcal{Z}^{-1} \exp\left(r_{\boldsymbol{\alpha}^*}(\boldsymbol{x})\right), \mathcal{Z} \in \mathbb{R}_+, \boldsymbol{\alpha}^* \in \mathbb{R}^d$. Then $\mathcal{Q}^*$ is the solution to

$$\arg\min_{Q \in \mathcal{S}} D_{KL}\left(Q \parallel P\right) \tag{16}$$

Moreover, $\mathcal{Q}^*$ is unique up to P-null sets.

Proof. Let $Q \in \mathcal{S}$ be an arbitrary measure, and denote the densities of Q and Q^* with respect to P by $f(\boldsymbol{x}) = dQ/dP$ and $f^*(\boldsymbol{x}) = dQ^*/dP$, respectively. Then

$$\mathrm{D}_{\mathrm{KL}}\left(Q^* \parallel P\right) = \int f^*(\boldsymbol{x}) \log f^*(\boldsymbol{x}) dP = \int f^*(\boldsymbol{x}) \{(\boldsymbol{\alpha}^*)^\mathsf{T} \boldsymbol{h}(\boldsymbol{x}) - \log \mathcal{Z}\} dP = (\boldsymbol{\alpha}^*)^\mathsf{T} \boldsymbol{h}^* - \log \mathcal{Z} = \int f(\boldsymbol{x}) \log f^*(\boldsymbol{x}) dP.$$

Then decomposing $\mathrm{D}_{\mathrm{KL}}\left(Q \parallel P\right)$ and substituting in our above expression yields

$$\begin{aligned}
\mathrm{D}_{\mathrm{KL}}\left(Q \parallel P\right) &\overset{(\star)}{=} \int f(\boldsymbol{x}) \log f^*(\boldsymbol{x}) dP + \int f(\boldsymbol{x}) \log \frac{f(\boldsymbol{x})}{f^*(\boldsymbol{x})} dP \\
&= \mathrm{D}_{\mathrm{KL}}\left(Q^* \parallel P\right) + \underbrace{\int f(\boldsymbol{x}) \log \frac{f(\boldsymbol{x})}{f^*(\boldsymbol{x})} dP}_{(\star\star)}.
\end{aligned}$$

In equality $(\star)$, we were allowed to add and subtract $\int f(\boldsymbol{x}) \log f^*(\boldsymbol{x}) dP$ since it is finite (and equal to $(\boldsymbol{\alpha}^*)^\mathsf{T} \boldsymbol{h}^* - \log \mathcal{Z}$ from above). And so it suffices to show $(\star\star)$ is nonnegative with equality if and only if $f(\boldsymbol{x}) = f^*(\boldsymbol{x})$ P-a.e.. Since f^* is strictly positive a.e. P, then Q^* and P are mutually absolutely continuous, and Q has a density with respect to Q^*. By the chain rule for Radon-Nikodym derivatives,

$$f(\boldsymbol{x}) = \frac{dQ}{dQ^*} f^*(\boldsymbol{x}) \; P\text{-a.e.}.$$

Substituting this result into $(\star\star)$, we obtain

$$\mathrm{D}_{\mathrm{KL}}\left(Q \parallel P\right) = \mathrm{D}_{\mathrm{KL}}\left(Q \parallel Q^*\right) + \mathrm{D}_{\mathrm{KL}}\left(Q^* \parallel P\right). \tag{17}$$

Since $\mathrm{D}_{\mathrm{KL}}\left(Q \parallel Q^*\right) \geq 0$ with equality iff $f(\boldsymbol{x}) = f^*(\boldsymbol{x})$ P-a.e., the result follows. □

Notably, since Theorem B.1 is stated for a general (measurable) space, it applies to Euclidean space $\mathbb{R}^d$ as well as to the space of continuous paths on $[0, 1]$ (endowed with the supremum norm). Once again, we point out the constraint set $\mathcal{S}$ of the maximum entropy problem (5) is strictly larger than that of the calibration problem (1), since it consists of all probability distributions defined on X that have a density with respect to P and satisfy the calibration constraint, some of which lie outside the family p_θ. Consequently, the optimum of (5) will be strictly smaller than that of the calibration problem (1). Specifically, equation (17) tells us that the gap between the calibration and KL solution is $\mathrm{D}_{\mathrm{KL}}\left(p_{\theta^*} \parallel p_{\theta_{\text{base}}}\right) - \mathrm{D}_{\mathrm{KL}}\left(p_{\boldsymbol{\alpha}^*} \parallel p_{\theta_{\text{base}}}\right) = \mathrm{D}_{\mathrm{KL}}\left(p_{\theta^*} \parallel p_{\boldsymbol{\alpha}^*}\right)$.

[1] For the case of a general distribution $p_{\theta'}$, the integrability conditions with respect to p_θ stated at the beginning of the section must be modified to integrability conditions with respect to $p_{\theta'}$.

B.1. Estimating the Maximum Entropy Solution

Next, we discuss our estimator (8) for the parameters $\boldsymbol{\alpha}^*$ of the reward $r_{\boldsymbol{\alpha}^*}$ and how it relates to the maximum entropy problem. First, we make the important observation that the constraint set $\mathcal{S}$ of the maximum entropy problem is convex (as a subset of the space of measures on X), as is the objective. And so, via convex duality (Wainwright et al., 2008, Theorem 3.4), solving the constrained maximum entropy problem over the space of probability distributions (16) is equivalent to solving the unconstrained problem in Euclidean space

$$\arg\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log\left(\int \exp\{r_{\boldsymbol{\alpha}}(\boldsymbol{x})\}dP\right), \tag{18}$$

where the objective can potentially take on the value $-\infty$ if $\log(\int \exp\{r_{\boldsymbol{\alpha}}(\boldsymbol{x})\}dP) = \infty$. This is the same as saying strong duality holds for the maximum entropy problem (16) and its dual (18). We detail the assumptions necessary to invoke strong duality in the formal statement of Proposition B.2. Moreover, the optimal value of (18) coincides with the KL distance of the maximum entropy solution (16) to p_θ. We verify in our proof of proposition B.2 that the objective (18) is strictly concave under weak assumptions.

Since the integral in (18) is typically intractable, the estimator we propose in (8) involves first sampling from the base model p_θ and then solving (18), with the integral replaced by the empirical average from our samples. The estimator $\widehat{\boldsymbol{\alpha}}$ is well-defined if and only if (i) $\boldsymbol{h}^*$ lies in the convex hull of $\{\boldsymbol{h}(\boldsymbol{x}_m)\}_{m=1}^M$ and (ii) the empirical covariance matrix of $\{\boldsymbol{h}(\boldsymbol{x}_m)\}_{m=1}^M$ has full rank. If (i) does not hold (but (ii) does), then the primal problem is infeasible. This means that although the objective (18) is strictly concave, the objective grows without bound to $+\infty$. Conversely, if (ii) does not hold (but (i) does), then there will exist infinitely many solutions to (8). However, under the assumptions of Proposition B.2, as $M \to \infty$, the probability that either (i) or (ii) does not hold approaches zero. Intuitively, since for each fixed $\boldsymbol{\alpha}$, the empirical averages approach the integral with probability one, we would expect that the solution to (16) would approach the solution to (18). We formalize this intuition in the following proposition:

Proposition B.2. *Assume that the components of $\boldsymbol{h}(\boldsymbol{x})$ are linearly independent as vectors in $\mathcal{L}^1(p_{\theta_{base}})$. Assume also for all $\boldsymbol{\alpha} \in \mathbb{R}^N$ belonging to an open neighborhood $N(\boldsymbol{\alpha}^*)$ of $\boldsymbol{\alpha}^*$,*

$$\int \exp\left\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\right\} p_{\theta_{base}}(\boldsymbol{x})d\boldsymbol{x} < \infty.$$

Let $\{\boldsymbol{x}_m\}_{m=1}^M$ be i.i.d. samples from the base generative model $p_{\theta_{base}}$. Then for any $\epsilon > 0$,

$$\mathbb{P}_{p_{\theta_{base}}}(\|\widehat{\boldsymbol{\alpha}}(\{\boldsymbol{x}_m\}_{m=1}^M) - \boldsymbol{\alpha}^*\| > \epsilon) \to 0 \quad \textit{as } M \to \infty.$$

In other words, the estimator $\widehat{\boldsymbol{\alpha}}$ is consistent for $\boldsymbol{\alpha}^*$.

Proof. Our argument relies upon the result that, under the assumptions stated in the proposition, $\boldsymbol{\alpha}^*$ is also the unique solution of the dual problem (18) (Wainwright et al., 2008, Theorem 3.4).

Our proof will proceed as follows: we first demonstrate that on a closed, bounded subset of $N(\boldsymbol{\alpha}^*)$ containing $\boldsymbol{\alpha}^*$, the approximation $\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - M^{-1}\sum_{m=1}^M \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x}_m)\}$ to the dual objective $\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log(\int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\}p_{\theta_{\text{base}}}(\boldsymbol{x})d\boldsymbol{x})$ converges uniformly with probability one. Then, since the dual objective is strictly concave and achieves its unique maximum at $\boldsymbol{\alpha}^*$, with high probability, the minimizer to the approximate objective will be close to $\boldsymbol{\alpha}^*$.

First, we recognize that the dual objective is a strictly concave function on $N(\boldsymbol{\alpha}^*)$. Indeed, it is straightforward to show

$$\nabla^2_{\boldsymbol{\alpha}} \log\left(\int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\}p_{\theta_{\text{base}}}(\boldsymbol{x})d\boldsymbol{x}\right) = \frac{\int (\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{m}_{\boldsymbol{\alpha}})(\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{m}_{\boldsymbol{\alpha}})^\mathsf{T} \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\}p_{\theta_{\text{base}}}(\boldsymbol{x})d\boldsymbol{x}}{\int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\}p_{\theta_{\text{base}}}(\boldsymbol{x})d\boldsymbol{x}} \succ 0$$

$$\boldsymbol{m}_{\boldsymbol{\alpha}} = \frac{\int \boldsymbol{h}(\boldsymbol{x}) \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\}p_{\theta_{\text{base}}}(\boldsymbol{x})d\boldsymbol{x}}{\int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\}p_{\theta_{\text{base}}}(\boldsymbol{x})d\boldsymbol{x}}.$$

Since $\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^*$ is concave in $\boldsymbol{\alpha}$, this implies that the dual objective is strictly concave on $N(\boldsymbol{\alpha}^*)$. Likewise, since the composition of a linear and convex function is convex, and log-sum-exp is a convex function, then the approximation to the dual objective is also concave in $\boldsymbol{\alpha} \in \mathbb{R}^N$, for each fixed $\boldsymbol{x} \in \mathbb{R}^d$.

Next, by the Strong Law of Large Numbers (SLLN), we can construct a Borel set $\widetilde{N} \subset \mathbb{R}^d$ such that $\widetilde{N}$ has probability zero under $p_{\theta_{\text{base}}}$ and on its complement,

$$\frac{1}{M}\sum_{m=1}^{M} \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x}_m)\} \to \int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\} p_{\theta_{\text{base}}}(\boldsymbol{x}) d\boldsymbol{x}$$

holds for each $\boldsymbol{\alpha} \in N(\boldsymbol{\alpha}^*) \cap \mathbb{Q}^N$ (apply SLLN for an individual $\boldsymbol{\alpha} \in N(\boldsymbol{\alpha}^*) \cap \mathbb{Q}^N$, then take a union over probability zero sets). Hence, on the complement of $\widetilde{N}$,

$$\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log\left(\frac{1}{M}\sum_{m=1}^{M} \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x}_m)\}\right) \to \boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log\left(\int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\} p_{\theta_{\text{base}}}(\boldsymbol{x}) d\boldsymbol{x}\right)$$

holds for each $\boldsymbol{\alpha} \in N(\boldsymbol{\alpha}^*) \cap \mathbb{Q}^N$.

A classical result in convex analysis (Rockafellar, 1997, Theorem 10.8) states that if a sequence of finite concave functions defined on an open, convex subset $\mathcal{C}$ converges pointwise on a dense subset of $\mathcal{C}$ to a limiting function, then the limiting function is concave on $\mathcal{C}$, and the convergence is uniform on closed and bounded subsets of $\mathcal{C}$. Applying this result to our setting, we have that on the complement of $\widetilde{N}$

$$\sup_{\boldsymbol{\alpha}\in K}\left|\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \frac{1}{M}\sum_{m=1}^{M} \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x}_m)\} - \left\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log\left(\int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\} p_{\theta_{\text{base}}}(\boldsymbol{x}) d\boldsymbol{x}\right)\right\}\right| \to 0$$

for K a closed and bounded subset of $N(\boldsymbol{\alpha}^*)$. Note that K can be chosen to have positive diameter since $N(\boldsymbol{\alpha}^*)$ is open.

Fix $\epsilon > 0$ sufficiently small such that the Euclidean ball centered at $\boldsymbol{\alpha}^*$ of radius ϵ is contained in K. We claim there exists some $\kappa \in \mathbb{R}$ such that for all $\|\boldsymbol{\alpha} - \boldsymbol{\alpha}^*\| = \epsilon$

$$\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \log\left(\int \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\} p_{\theta_{\text{base}}}(\boldsymbol{x}) d\boldsymbol{x}\right) < \kappa < (\boldsymbol{\alpha}^*)^\mathsf{T}\boldsymbol{h}^* - \log\left(\int \exp\{(\boldsymbol{\alpha}^*)^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\} p_{\theta_{\text{base}}}(\boldsymbol{x}) d\boldsymbol{x}\right).$$

The existence of such a $\kappa \in \mathbb{R}$ follows from (i) the fact that the left-hand side of the above inequality attains its maximum on the compact set $\|\boldsymbol{\alpha} - \boldsymbol{\alpha}^*\| = \epsilon$ and (ii) by strict concavity of the dual objective, this maximum must be strictly less than the right-hand side. Fix $\delta > 0$. By the uniform convergence previous proved, there exists $M_{\epsilon,\delta} \in \mathbb{N}$ such that $\forall M \geq M_{\epsilon,\delta}$ and for all $\|\boldsymbol{\alpha} - \boldsymbol{\alpha}^*\| = \epsilon$,

$$\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}^* - \frac{1}{M}\sum_{m=1}^{M} \exp\{\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x}_m)\} < (\boldsymbol{\alpha}^*)^\mathsf{T}\boldsymbol{h}^* - \frac{1}{M}\sum_{m=1}^{M} \exp\{(\boldsymbol{\alpha}^*)^\mathsf{T}\boldsymbol{h}(\boldsymbol{x}_m)\}$$

with probability at least $1 - \delta$ under $p_{\theta_{\text{base}}}$. But since the approximate dual objective is concave, this implies that, on this event, its maximum occurs within the Euclidean ball of radius ϵ.

In other words, we have proven that for every $\epsilon > 0, \delta > 0$, there exists $M_{\epsilon,\delta}$ such that for every $M \geq M_{\epsilon,\delta}$,

$$\mathbb{P}_{p_{\theta_{\text{base}}}}\left(\|\widehat{\boldsymbol{\alpha}}(\{\boldsymbol{x}_m\}_{m=1}^M) - \boldsymbol{\alpha}^*\| > \epsilon\right) \leq \delta.$$

□

C. Neural Stochastic Differential Equations

Consider the measurable space $(C[0,1]^d, \mathcal{B}_{C[0,1]^d})$ of d-dimensional continuous paths on the interval $0 \leq t \leq 1$, where $\mathcal{B}_{C[0,1]^d}$ is the Borel sigma-field corresponding to the supremum norm on $C[0,1]^d$. Next, we discuss neural stochastic differential equations, which model $(\boldsymbol{x}(t))_{0\leq t\leq 1}$ as the solution to a stochastic differential equation

$$d\boldsymbol{x}(t) = b_\theta(\boldsymbol{x}(t), t)dt + \sigma(t)d\boldsymbol{w}(t),\ \boldsymbol{x}(0) \sim p_{\text{init}}, \tag{19}$$

where $(\boldsymbol{w}(t))_{0\leq t\leq 1}$ is a standard d-dimensional Brownian motion, b_θ is modeled by a neural network, and p_{init} is a known distribution from which sampling is tractable. When the drift $b_\theta : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ and diffusion $\sigma : [0,1] \to \mathbb{R}_+$

coefficients are bounded and b_θ satisfies the Lipschitz continuity condition $\|b_\theta(\boldsymbol{x}, t) - b_\theta(\boldsymbol{y}, t)\| \leq C_1(1 + \|\boldsymbol{x} - \boldsymbol{y}\|)$ for some $C_1 \in \mathbb{R}_+$ and all $t \in [0, 1],\ \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, the SDE (19) admits a unique, strong solution (Oksendal, 2013, Theorem 5.2.1). We denote the solution, which is a probability law on $(C[0,1]^d, \mathcal{B}_{C[0,1]^d})$, by p_θ.

We will be interested in solving the calibration problem when the base model $p_{\theta_{\text{base}}}$ and calibrated model p_θ are neural-SDEs. For our proof of Theorem 4.1, we will also need to assume that the SDE (19) is uniformly elliptic, meaning $\sigma(t) > \kappa > 0$ for all $0 \leq t \leq 1$ and some $\kappa > 0$, and that $b_{\theta_{\text{base}}}(\boldsymbol{x}, t)$, $\sigma(t)$ are uniformly Lipschitz on $\mathbb{R}^d \times [0, 1]$ (rather than for each fixed t). It will be convenient to represent the calibrated model as a *controlled diffusion process*

$$\begin{aligned} p_\theta : d\boldsymbol{x}(t) &= (b_{\theta_{\text{base}}}(\boldsymbol{x}(t), t) + \sigma(t) u_\theta(\boldsymbol{x}(t), t))dt + \sigma(t) d\boldsymbol{w}(t),\ \boldsymbol{x}(0) \sim p_{\text{init}}, \\ u_\theta(\boldsymbol{x}(t), t) &= \sigma(t)^{-1}(b_\theta(\boldsymbol{x}(t), t) - b_{\theta_{\text{base}}}(\boldsymbol{x}(t), t)). \end{aligned} \tag{20}$$

In this formulation $u_\theta(\boldsymbol{x}(t), t)$ is called the *controller*. Intuitively, an identically zero controller $u_\theta = 0$ recovers the base model p_θ whereas a nonzero controller perturbs the generative model away from the base model.

An important consequence of our assumptions is Girsanov's Theorem (Cameron & Martin, 1944; Girsanov, 1960), which tells us that the probability measures p_θ and $p_{\theta_{\text{base}}}$ have densities with respect to one another.

Theorem C.1 (Girsanov's Theorem)**.** *Suppose the SDEs*

$$\begin{aligned} \nu_1(\boldsymbol{x}) &: d\boldsymbol{x}(t) = b_1(\boldsymbol{x}(t), t)dt + \sigma(t)d\boldsymbol{w}(t), \quad 0 \leq t \leq 1 \\ \nu_2(\boldsymbol{x}) &: d\boldsymbol{x}(t) = (b_1(\boldsymbol{x}(t), t) + \sigma(t) b_2(\boldsymbol{x}(t), t))dt + \sigma(t)d\boldsymbol{w}(t), \quad 0 \leq t \leq 1 \end{aligned}$$

satisfy $\sigma(t) > 0,\ 0 < t < 1$, have the same initial law $\nu_1(\boldsymbol{x}_0) = \nu_2(\boldsymbol{x}_0)$, and admit unique, strong solutions, ν_1 and ν_2. Suppose also

$$\left[\frac{\nu_2(\boldsymbol{x})}{\nu_1(\boldsymbol{x})}\right]_t = \exp\left\{\sum_{i=1}^d \int_0^t b_2(\boldsymbol{x}(t), t)_i d\boldsymbol{w}_i^{\nu_1}(t) - \frac{1}{2}\int_0^t \|b_2(\boldsymbol{x}(t), t)\|^2 dt\right\}$$

is a ν_1-martingale, where $(\boldsymbol{w}^{\nu_1}(t))_{0 \leq t \leq 1}$ is a ν_1-Brownian motion and $d\boldsymbol{w}_i^{\nu_1}(t),\ i = 1, \ldots, d$ denotes the Itô stochastic integral. Then the probability measure ν_2 has a density with respect to ν_1. In particular, for any bounded functional Φ defined on $C[0,1]^d$,

$$\mathbb{E}_{\nu_2}[\Phi(\boldsymbol{x})] = \mathbb{E}_{\nu_1}\left[\Phi(\boldsymbol{x})\left[\frac{\nu_2(\boldsymbol{x})}{\nu_1(\boldsymbol{x})}\right]_1\right].$$

Under our assumptions, $([p_\theta(\boldsymbol{x})/p_{\theta_{\text{base}}}(\boldsymbol{x})]_t)_{0 \leq t \leq 1}$ is a martingale with respect to $p_{\theta_{\text{base}}}$. Consequently, Girsanov's Theorem implies that the density of the controlled diffusion process p_θ with respect to $p_{\theta_{\text{base}}}$ is given by

$$\frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} := \left[\frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})}\right]_1 = \exp\left\{\sum_{i=1}^d \int_0^1 u_\theta(\boldsymbol{x}(t), t)_i d\boldsymbol{w}_i^{p_{\theta_{\text{base}}}}(t) - \frac{1}{2}\int_0^1 \|u_\theta(\boldsymbol{x}(t), t)\|^2 dt\right\}, \tag{21}$$

This expression for the density of p_θ with respect to $p_{\theta_{\text{base}}}$ allows us to compute the KL divergence between the probability measures p_θ and $p_{\theta_{\text{base}}}$ according to

$$\mathrm{D}_{\text{KL}}\left(p_\theta \parallel p_{\theta_{\text{base}}}\right) = \frac{1}{2}\int_0^1 \mathbb{E}_{p_\theta}\|u_\theta(\boldsymbol{x}(t), t)\|^2 dt.$$

The stochastic integral term vanishes since it has expectation zero.

We point out that we have been notationally imprecise insofar as $p_\theta(\boldsymbol{x})$ and $p_{\theta_{\text{base}}}(\boldsymbol{x})$ represent probability measures on the space of d-dimensional continuous paths, rather than densities with respect to a measure on d-dimensional Euclidean space. Still, Girsanov's Theorem tells us how to compute the density of p_θ with respect to $p_{\theta_{\text{base}}}$. The importance weights $\frac{p_\theta(\boldsymbol{x})}{p_{\texttt{stop-grad}(\theta)}(\boldsymbol{x})}$ can likewise be computed via Girsanov's Theorem. We will demonstrate in Appendix C.2 that these are exactly the two quantities we need in order to compute our unbiased gradient estimates for $\mathcal{L}^{\text{relax}}$ and $\mathcal{L}^{\text{reward}}$.

C.1. Characterization of Maximum Entropy Solution

Next, we address why optimizing the reward loss $\mathcal{L}^{\text{reward}}$ approximately solves the calibration problem, which aims to find the controller for which p_θ is closest in KL distance to $p_{\theta_{\text{base}}}$. In particular, we demonstrate that when the calibration constraint is a function of the terminal state $\boldsymbol{x}(1)$ only, and when $\boldsymbol{x}(0) \perp \boldsymbol{x}(1)$ are independent under $p_{\theta_{\text{base}}}$, then the solution to the maximum entropy problem (5) is, in fact, a controlled diffusion process of the form (20). Moreover, we give a closed-form expression for the optimal controller in this case.

proof of Theorem 4.1. By Theorem B.1, the solution to the maximum entropy problem (16) has density

$$\frac{p_{\boldsymbol{\alpha}^*}(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} = \frac{\exp\{r_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))\}}{\mathbb{E}_{p_{\theta_{\text{base}}}}[\exp\{r_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))\}]} \tag{22}$$

with respect to $p_{\theta_{\text{base}}}$. The denominator here is a normalizing constant i.e. it ensures the probability density integrates to one. We point out that by the properties of conditional expectation (Léonard, 2014),

$$\frac{p_{\boldsymbol{\alpha}^*}(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} = \frac{p_{\boldsymbol{\alpha}^*}(\cdot|\boldsymbol{x}(1))}{p_{\theta_{\text{base}}}(\cdot|\boldsymbol{x}(1))} \frac{p_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))}{p_{\theta_{\text{base}}}(\boldsymbol{x}(1))},\ p_{\theta_{\text{base}}}\text{-a.e.} \quad \text{where} \quad \frac{p_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))}{p_{\theta_{\text{base}}}(\boldsymbol{x}(1))} = \mathbb{E}_{p_{\theta_{\text{base}}}}\left[\frac{p_{\boldsymbol{\alpha}^*}(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})}\middle|\boldsymbol{x}(1)\right],\ p_{\theta_{\text{base}}}\text{-a.e..} \tag{23}$$

However, since the density of $p_{\boldsymbol{\alpha}^*}$ is a function of $\boldsymbol{x}(1)$ only, this implies $\frac{p_{\boldsymbol{\alpha}^*}(\cdot|\boldsymbol{x}(1))}{p_{\theta_{\text{base}}}(\cdot|\boldsymbol{x}(1))} = 1$, $p_{\theta_{\text{base}}}$-a.e. In other words, to obtain the distribution $p_{\boldsymbol{\alpha}^*}$, one only needs to reweight the paths from $p_{\theta_{\text{base}}}$ according to their terminal value.

From here, we proceed by determining a controller u for which the solution to the controlled diffusion process (20) is equal in law to the solution of the maximum entropy problem (22). Within the stochastic differential equation literature, our proof technique is referred to as the Doob h-transform argument (Karatzas & Shreve, 2012; Oksendal, 2013; Denker et al., 2024). To this end, consider the Backward Kolmogorov Equation (BKE): the solution $g \in C^{2,1}(\mathbb{R}^d \times [0,1])$ (i.e. twice continuously differentiable in $\boldsymbol{x}$, one continuously differentiable in t) to the BKE satisfies the partial differential equation

$$\partial_t g(\boldsymbol{x},t) + \frac{1}{2}\sigma^2(t)\Delta_{\boldsymbol{x}} g(\boldsymbol{x},t) + b_{\theta_{\text{base}}}(\boldsymbol{x},t)^\mathsf{T}\nabla_{\boldsymbol{x}} g(\boldsymbol{x},t) = 0, \quad g(\boldsymbol{x},1) = \exp\{r_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))\}, \quad t \in [0,1].$$

where, from our assumptions, $g(\boldsymbol{x},1)$ is bounded and continuous. Under the assumptions stated at the beginning of the section, there exists a unique solution to the BKE, which is given by

$$g(\boldsymbol{x},t) = \mathbb{E}_{p_{\theta_{\text{base}}}}[\exp\{r_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))\}|\boldsymbol{x}(t) = \boldsymbol{x}].$$

See Karatzas & Shreve (2012, Chapter 5) as well as Friedman (1975, Chapter 6). By Itô's Lemma applied to the function $g(\boldsymbol{x},t)$ and the base process $(\boldsymbol{x}(t))_{0\le t\le 1}$,

$$\begin{aligned}
dg(\boldsymbol{x}(t),t) &= \partial_t g(\boldsymbol{x}(t),t) + \sum_{i=1}^d (\nabla_{\boldsymbol{x}} g(\boldsymbol{x}(t),t))_i d\boldsymbol{x}(t)_i + \frac{1}{2}\sum_{i=1}^d (\nabla_{\boldsymbol{x}} g(\boldsymbol{x}(t),t))_{i,i} (d\boldsymbol{x}(t)_i)^2 \\
&= \underbrace{\left\{\partial_t g(\boldsymbol{x}(t),t) + \frac{1}{2}\sigma(t)^2\Delta_{\boldsymbol{x}} g(\boldsymbol{x}(t),t) + b_{\theta_{\text{base}}}(\boldsymbol{x}(t),t)^\mathsf{T}\nabla_{\boldsymbol{x}} g(\boldsymbol{x}(t),t)\right\} dt}_{=0} + \sigma(t)\sum_{i=1}^d (\nabla_{\boldsymbol{x}} g(\boldsymbol{x}(t),t))_i d\boldsymbol{w}(t)_i \\
&= \sigma(t)\sum_{i=1}^d (\nabla_{\boldsymbol{x}} g(\boldsymbol{x}(t),t))_i d\boldsymbol{w}(t)_i.
\end{aligned}$$

The dt term vanishes since g is the solution to the BKE. Since g is bounded, then the right-hand side, in addition to being a local martingale, is a bona fide martingale with respect to $p_{\theta_{\text{base}}}$.

Moreover, since $\log(z)$ is continuous on the set in which $g(\boldsymbol{x},t)$ is contained, we may invoke Ito's Lemma once more

$$\begin{aligned}
d\log g(\boldsymbol{x}(t),t) &= \frac{1}{g(\boldsymbol{x}(t),t)} dg(\boldsymbol{x}(t),t) - \frac{1}{2g(\boldsymbol{x}(t),t)^2}(dg(\boldsymbol{x}(t),t))^2 \\
&= \sigma(t)\sum_{i=1}^d (\nabla_{\boldsymbol{x}} \log g(\boldsymbol{x}(t),t))_i d\boldsymbol{w}(t)_i - \frac{1}{2}\left\|\sigma(t)\nabla_{\boldsymbol{x}} \log g(\boldsymbol{x}(t),t)\right\|_2^2 dt.
\end{aligned} \tag{24}$$

Exponentiating both sides yields

$$\frac{g(\boldsymbol{x}(t),t)}{g(\boldsymbol{x}(0),0)} = \exp\left\{\int_0^t \sum_{i=1}^d \sigma(s)(\nabla_{\boldsymbol{x}} \log g(\boldsymbol{x}(s),s))_i d\boldsymbol{w}(s)_i - \frac{1}{2}\int_0^t \left\|\sigma(s)\nabla_{\boldsymbol{x}} \log g(\boldsymbol{x}(s),s)\right\|^2 ds\right\}.$$

Again, since the left-hand side is bounded, then the right-hand side is a martingale with respect to $p_{\theta_{\text{base}}}$.

From here, we make two important observations. First, since $\boldsymbol{x}(0) \perp \boldsymbol{x}(1)$ under $p_{\theta_{\text{base}}}$, then

$$g(\boldsymbol{x}(0),0) = \mathbb{E}_{p_{\theta_{\text{base}}}}[\exp\{r_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))\}|\boldsymbol{x}(0)=\boldsymbol{x}] = \mathbb{E}_{p_{\theta_{\text{base}}}}[\exp\{r_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))\}]$$

is a constant with respect to $p_{\theta_{\text{base}}}$. This is exactly the normalizing constant appearing in $p_{\boldsymbol{\alpha}^*}$. In particular, if we evaluate the above expression (24) at time $t=1$ and plug in our expression for $g(\boldsymbol{x}(0),0)$, we obtain

$$\frac{p_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))}{p_{\theta_{\text{base}}}(\boldsymbol{x}(1))} = \frac{g(\boldsymbol{x}(1),1)}{g(\boldsymbol{x}(0),0)} = \exp\left\{\int_0^1 \sum_{i=1}^d \sigma(s)(\nabla_{\boldsymbol{x}} \log g(\boldsymbol{x}(s),s))_i d\boldsymbol{w}(s)_i - \frac{1}{2}\int_0^1 \left\|\sigma(s)\nabla_{\boldsymbol{x}} \log g(\boldsymbol{x}(s),s)\right\|^2 ds\right\}. \tag{25}$$

Second, by Girsanov's Theorem (Theorem C.1), (25) represents the density of the controlled SDE

$$p_{u^*} : d\boldsymbol{x}(t) = \{b_{\theta_{\text{base}}}(\boldsymbol{x}(t),t) + \sigma(t)u^*(\boldsymbol{x}(t),t)\}dt + \sigma(t)d\boldsymbol{w}(t),\ \boldsymbol{x}(0) \sim p_{\text{init}},\ u^* := \sigma(t)\nabla_{\boldsymbol{x}} \log g(\boldsymbol{x},t)$$

with respect to $p_{\theta_{\text{base}}}$. Combining these two facts, we obtain

$$\frac{p_{\boldsymbol{\alpha}^*}(\boldsymbol{x}(1))}{p_{\theta_{\text{base}}}(\boldsymbol{x}(1))} = \frac{p_{u^*}(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})},\ p_{\theta_{\text{base}}}\text{-a.e..}$$

Finally, by (23), we conclude

$$\frac{p_{\boldsymbol{\alpha}^*}(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} = \frac{p_{u^*}(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})},\ p_{\theta_{\text{base}}}\text{-a.e..}$$

Equivalently, $p_{\boldsymbol{\alpha}^*}$ is equal in law to the controlled diffusion process p_{u^*} with controller $u^* = \sigma(t)\nabla_{\boldsymbol{x}} \log g(\boldsymbol{x},t)$. □

C.2. CGM-relax and CGM-reward for Neural-SDEs

In this section, we discuss how we implement CGM-relax and CGM-reward in the setting of a neural-SDE pre-trained model and a controlled diffusion process calibrated model. As we mentioned in Appendix C, since $p_{\theta_{\text{base}}}$ and p_θ represent probability measures rather than densities, we cannot directly compute the gradient estimates we proposed for $\nabla_\gamma \mathcal{L}^{\text{relax}}$ and $\nabla_\gamma \mathcal{L}^{\text{reward}}$ in Appendix A.2.

We start with our gradient estimate for the reward loss (9). Define $\texttt{sg}(\cdot) := \texttt{stop-grad}(\cdot)$. In particular, rather than introducing the score, we can work with the importance weights $\frac{p_\theta(\boldsymbol{x})}{p_{\texttt{sg}(\theta)}(\boldsymbol{x})}$, which are well-defined via Girsanov's Theorem. The gradient of the reward loss is

$$\begin{aligned}
\nabla_\theta \mathcal{L}^{\text{reward}}(\theta) &= \nabla_\theta \mathbb{E}_{p_\theta}\left[\log \frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right] \\
&= \nabla_\theta \int \left\{\log \frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right\} p_\theta(d\boldsymbol{x}) \\
&\overset{(\star)}{=} \nabla_\theta \int \frac{p_\theta(\boldsymbol{x})}{p_{\texttt{sg}(\theta)}(\boldsymbol{x})}\left\{\log \frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right\} p_{\texttt{sg}(\theta)}(d\boldsymbol{x}) \\
&= \mathbb{E}_{p_{\texttt{sg}(\theta)}}\left[\left(\nabla_\theta \frac{p_\theta(\boldsymbol{x})}{p_{\texttt{sg}(\theta)}(\boldsymbol{x})}\right)\left\{\log \frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right\}\right] + \mathbb{E}_{p_\theta}\left[\nabla_\theta \log \frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})}\right].
\end{aligned}$$

In equality $(\star)$ we invoked Girsanov's Theorem to change measure from p_θ to $p_{\texttt{sg}(\theta)}$. Just as in Appendix A.2, the second term is zero. One can see this by rewriting

$$\frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} = \frac{p_\theta(\boldsymbol{x})}{p_{\texttt{sg}(\theta)}(\boldsymbol{x})}\frac{p_{\texttt{sg}(\theta)}(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})}\ p_\theta\text{-a.e..}$$

As for the first term, we have from Girsanov's Theorem

$$\nabla_\theta \mathcal{L}^{\text{reward}}(\theta) = \mathbb{E}_{p_{\text{sg}(\theta)}}\left[\left(\nabla_\theta \frac{p_\theta(\boldsymbol{x})}{p_{\text{sg}(\theta)}}\right)\left\{\log \frac{p_\theta(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right\}\right]$$
$$\overset{(\star)}{=} \mathbb{E}_{p_{\text{sg}(\theta)}}\left[\left(\sum_{i=1}^d \int_0^1 \nabla_\theta u_\theta(\boldsymbol{x}(t),t)_i d\boldsymbol{w}_i^{p_{\text{sg}(\theta)}}(t)\right)\left(\sum_{i=1}^d \int_0^1 u_\theta(\boldsymbol{x}(t),t)_i d\boldsymbol{w}_i^{p_{\theta_{\text{base}}}}(t) - \frac{1}{2}\int_0^1 \|u_\theta(\boldsymbol{x}(t),t)\|^2 dt - r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right)\right].$$

Here, $(\star)$ follows from differentiating the density of p_θ with respect to $p_{\text{sg}(\theta)}$:

$$\nabla_\theta \frac{p_\theta(\boldsymbol{x})}{p_{\text{sg}(\theta)}} = \nabla_\theta \exp\left\{\sum_{i=1}^d \int_0^1 \{u_\theta(\boldsymbol{x}(t),t)_i - u_{\text{sg}(\theta)}(\boldsymbol{x}(t),t)_i\} d\boldsymbol{w}_i^{p_{\text{sg}(\theta)}}(t) - \frac{1}{2}\int_0^1 \|u_\theta(\boldsymbol{x}(t),t) - u_{\text{sg}(\theta)}(\boldsymbol{x}(t),t)\|^2 dt\right\}$$
$$= \sum_{i=1}^d \int_0^1 \nabla_\theta u_\theta(\boldsymbol{x}(t),t)_i d\boldsymbol{w}_i^{p_{\text{sg}(\theta)}}(t).$$

Moreover, by Girsanov's Theorem, it holds

$$d\boldsymbol{w}_i^{p_{\theta_{\text{base}}}}(t) = d\boldsymbol{w}_i^{p_{\text{sg}(\theta)}}(t) + u_{\text{sg}(\theta)}(\boldsymbol{x}(t),t)dt,$$

which yields

$$\nabla_\theta \mathcal{L}^{\text{reward}}(\theta) = \mathbb{E}_{p_{\text{sg}(\theta)}}\left[\left(\sum_{i=1}^d \int_0^1 \nabla_\theta u_\theta(\boldsymbol{x}(t),t)_i d\boldsymbol{w}_i^{p_{\text{sg}(\theta)}}(t)\right)\widetilde{Y}_{\theta,\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})\right] \tag{26}$$
$$\widetilde{Y}_{\theta,\boldsymbol{\alpha}}(\boldsymbol{x}) := \sum_{i=1}^d \left(\int_0^1 u_\theta(\boldsymbol{x}(t),t)_i d\boldsymbol{w}_i^{p_{\text{sg}(\theta)}}(t) + \int_0^1 u_{\text{sg}(\theta)}(\boldsymbol{x}(t),t)_i u_\theta(\boldsymbol{x}(t),t)_i dt\right) - \frac{1}{2}\int_0^1 \|u_\theta(\boldsymbol{x}(t),t)\|^2 dt - r_{\boldsymbol{\alpha}}(\boldsymbol{x}).$$

We compute an unbiased estimate to (26) by drawing M *i.i.d.* samples $\{\boldsymbol{x}_m\}_{m=1}^M$ from $p_{\text{sg}(\theta)}$ and computing a Monte Carlo estimate. When sampling from $p_{\text{sg}(\theta)}$ according to an SDE solver (e.g., Euler-Maruyama), we also store the Brownian motion that we sample $\{\boldsymbol{w}_m^{p_{\text{sg}(\theta)}}\}_{m=1}^M$. Just as we did in Appendix A.2, we can also include a baseline to reduce the variance of our gradient estimate. Altogether, we get the following unbiased estimate for $\nabla_\theta \mathcal{L}^{\text{reward}}$:

$$\frac{1}{M}\sum_{m=1}^M \left\{\left(\sum_{i=1}^d \int_0^1 \nabla_\theta u_\theta(\boldsymbol{x}_m(t),t)_i d\boldsymbol{w}_{m,i}^{p_{\text{sg}(\theta)}}(t)\right)\left(\widetilde{Y}_{\theta,\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_m) - \frac{1}{M-1}\sum_{m'\neq m}\widetilde{Y}_{\theta,\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_{m'})\right)\right\}. \tag{27}$$

As pointed out by Richter et al. (2020), our gradient estimator (27)—the score-based gradient estimator with a baseline—coincides up to a constant multiplicative factor with the gradient estimate for the log-variance loss (Nüsken & Richter, 2021). The log-variance loss is defined as

$$\text{Var}_{p_{\text{sg}(\theta)}}\left(\log \frac{p_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})}{p_\theta(\boldsymbol{x})}\right),$$

where $p_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})$ is defined such that $\frac{p_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})}{p_{\theta_{\text{base}}}(\boldsymbol{x})} = \mathcal{Z}^{-1}\exp\{\widehat{\boldsymbol{\alpha}}^\mathsf{T}\boldsymbol{h}(\boldsymbol{x})\}$ (this is also we defined $p_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x})$ in the proof of Theorem 4.1). In practice, one estimates the log-variance loss by computing

$$\frac{1}{M-1}\sum_{m=1}^M \left(\widetilde{Y}_{\theta,\boldsymbol{\alpha}}(\boldsymbol{x}_m) - \frac{1}{M}\sum_{m'=1}^M \widetilde{Y}_{\theta,\boldsymbol{\alpha}}(\boldsymbol{x}_{m'})\right)^2$$

for $\{\boldsymbol{x}_m\}_{m=1}^M$ drawn *i.i.d.* from $p_{\text{sg}(\theta)}$. And by differentiating this expression, one obtains an unbiased gradient estimate for the log-variance loss.

As for the relaxed loss (2), we can estimate the gradient of the KL term in an identical way i.e. by computing

$$\frac{1}{M}\sum_{m=1}^M \left\{\left(\sum_{i=1}^d \int_0^1 \nabla_\theta u_\theta(\boldsymbol{x}_m(t),t)_i d\boldsymbol{w}_{m,i}^{p_{\text{sg}(\theta)}}(t)\right)\left(\widetilde{Y}_{\theta,\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_m) + r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_m) - \frac{1}{M-1}\sum_{m'\neq m}\left\{\widetilde{Y}_{\theta,\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_{m'}) + r_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_{m'})\right\}\right)\right\}.$$

And for the squared norm difference between the expectation of $\boldsymbol{h}$ under p_θ and the target expectation $\boldsymbol{h}^*$, we use the same estimate $\nabla_\theta \widehat{\mathcal{L}}^{\text{viol}}(\{\frac{p_\theta(\boldsymbol{x}_m)}{p_{\text{sg}(\theta)}(\boldsymbol{x}_m)}\tilde{\boldsymbol{h}}_m\}_{m=1}^M)$ as in Appendix A.2. Now, though, the density $\frac{p_\theta(\boldsymbol{x}_m)}{p_{\text{sg}(\theta)}(\boldsymbol{x}_m)}$ is determined by Girsanov's Theorem (Theorem C.1).

D. Additional Experimental Details

D.1. Synthetic Data Experiments

For each of our synthetic data experiments, we represent $p_{\theta_{\text{base}}}$ as the solution to

$$p_{\theta_{\text{base}}} : d\boldsymbol{x}(t) = \underbrace{\{\sigma(t)^2 s(\boldsymbol{x}(t), 1-t) - f(\boldsymbol{x}(t), 1-t)\}}_{b_{\theta_{\text{base}}}} dt + \sigma(t) d\boldsymbol{w}(t),\ 0 \le t \le 1,\ \boldsymbol{x}(0) \sim p_{\text{init}},$$

where $s(\boldsymbol{x}(t), t) = \nabla_{\boldsymbol{x}} \log \overrightarrow{p}(\boldsymbol{x}(t))$ is the score of the forward noising process defined by drift $f(\boldsymbol{x}, t)$ and diffusion $\sigma(1-t)$. The forward process satisfies the SDE

$$\overrightarrow{p} : d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), t)dt + \sigma(1-t)d\boldsymbol{w}(t),\ \boldsymbol{x}(0) \overset{d}{=} p_{\theta_{\text{base}}}(\boldsymbol{x}(1)).$$

When $f(\boldsymbol{x}(t), t) = \kappa_t \boldsymbol{x}(t)$ for some $(\kappa_t)_{0 \le t \le 1}$ (i.e. the forward noising process is defined by a linear SDE / Ornstein–Uhlenbeck process) and $\overrightarrow{p}(\boldsymbol{x}(0))$ is a Gaussian mixture model, $s(\boldsymbol{x}(t), t)$ admits a closed form. See Song et al. (2020) for further details on diffusion processes and their reversals.

We parameterize the drift of the calibrated model as $b_{\theta_{\text{base}}}(\boldsymbol{x}, t) + \sigma(t) u_\theta(\boldsymbol{x}, t)$, where $u_\theta(\boldsymbol{x}, t)$ is a neural network with two hidden layers of dimension 256 with SiLU activations. In addition to $\boldsymbol{x}(t)$, we also input to the neural network a sinusoidal time embedding of dimension 32.

When performing our synthetic data experiments with the CGM-reward algorithm, we estimate the parameters $\boldsymbol{\alpha}^*$ of the maximum entropy solution (6) using 10^4 samples from the base model p_θ. In the Gaussian mean experiments, where the dimension of $\boldsymbol{\alpha}$ is equal to the dimension of the problem, we observe that the error in estimating $\boldsymbol{\alpha}^*$ grows with the dimensionality of the problem. Theoretically characterizing the dependence of the estimator $\widehat{\boldsymbol{\alpha}}$ on the dimension of the constraint remains future work.

For each of our synthetic data experiments, we perform the CGM updates described in Algorithms 1 and 2 using Adam with a cosine learning rate schedule. For the GMM experiments we perform $2.5 \cdot 10^3$ iterations and for the Gaussian mean experiments we perform 10^3 iterations. These values were chosen by inspecting each term of the relax and reward losses (see equations (2) and (9)). For each set of experiments, we assessed the stability and convergence speed of the CGM algorithm for various values of minimum and maximum learning rate. Namely, for the GMM experiments we chose maximum learning rate 10^{-3} and minimum learning rate 10^{-5} for both CGM-relax and CGM-reward. For the Gaussian moment calibration experiments, we chose maximum learning rate 10^{-4} for both algorithms and minimum learning rates 10^{-6} and 10^{-7} for CGM-relax and CGM-reward, respectively.

In both synthetic data experiments, we use $M = 10^4$ samples to perform each gradient update (evaluating the neural network is inexpensive for these problems). Samples are drawn on a time-grid of size 100.

D.2. Calibrating the Genie2 Protein Design Model

For our protein diffusion model experiments, we use the Genie2 encoder-decoder architecture to define $p_{\theta_{\text{base}}}$ (see details in Lin et al. (2024)), which is constructed to be SE(3)-equivariant. We then directly fine-tune the base Genie model by initializing p_θ at $p_{\theta_{\text{base}}}$.

Since Genie2 is trained as the reversal of a discrete-time noising process (a DDPM, see Ho et al., 2020), we first convert the discrete-time denoising diffusion model to a (continuous-time) neural-SDE. We do this by redefining the final timestep T of the original denoising process to be time 1 of the continuous-time process. To define the drift function, we take the DDPM transition mean defined at each time t in the discrete-time process, divide it by $1/T = T$, and define the drift function to be equal to the resulting value in between times t/T and $(t+1)/T$. The diffusion coefficient is similarly defined by the DDPM transition standard deviation at each time t in the discrete-time process, but is instead scaled by $T^{1/2}$. This approach of converting the DDPM into a neural-SDE ensures that when the SDE is solved under the Euler-Maruyama scheme using a grid of T timesteps (i.e. the original time grid used to define the DDPM), one samples from the original DDPM.

Again for the Genie2 experiments, we perform the CGM-relax updates using Adam with a cosine learning rate schedule. The calibrated models displayed in Figure 2 and Figure 4 are each trained with maximum learning rate 10^{-5} and minimum learning rate 10^{-7}. Across all experiments, we run 100 CGM-relax training iterations, each of which samples $M = 64$ structures to perform the gradient update. Gradient computation is parallelized across 4 NVIDIA H100 GPUs. In future work, we intend to consider alternative parameterizations of the controller in order to make generating samples with Genie2 more memory efficient.

For Genie2, we perform sampling using 100 timesteps and a non-uniform time grid: we sample the first 50 steps on the interval $[0, 0.05]$ and the remaining steps on the interval $[0.05, 1]$. We point out that the original Genie2 model was trained with 10^3 denoising steps; reducing the number of sampling steps substantially decreases the runtime of CGM calibration. Our sampling scheme is possible since we redefined the base generative model to be a neural-SDE. From computing self-consistency metrics for the base Genie2 model sampled on the original time grid (with 10^3 steps) and on our proposed grid (with 100 steps), we did not observe any difference in sample quality.

Self-consistency RMSD and designability. To assess the quality of our generations, we compute the root mean-square deviation (RMSD) between C_α atoms resulting from unfolding our generated structures into predicted amino sequences and then refolding each of these predicted sequences into a protein structure. The self-consistency RMSD (scRMSD) is defined as the smallest RMSD between the given structure and one of the corresponding predictions. We use ProteinMPNN (Dauparas et al., 2022) for our inverse folding model and ESMFold (Lin et al., 2023) for our folding model; we compute scRMSD from 8 sequences. The pipeline we employ was developed by Lin & Nguyen (2024). Once we have determined the scRMSD of a generated structure, we classify it as "designable" if its scRMSD is at most 2Å. Intuitively, designability is a binary measure of whether or not a structure could have been plausibly produced by folding an amino acid sequence.

D.3. Additional Figures from Experiments

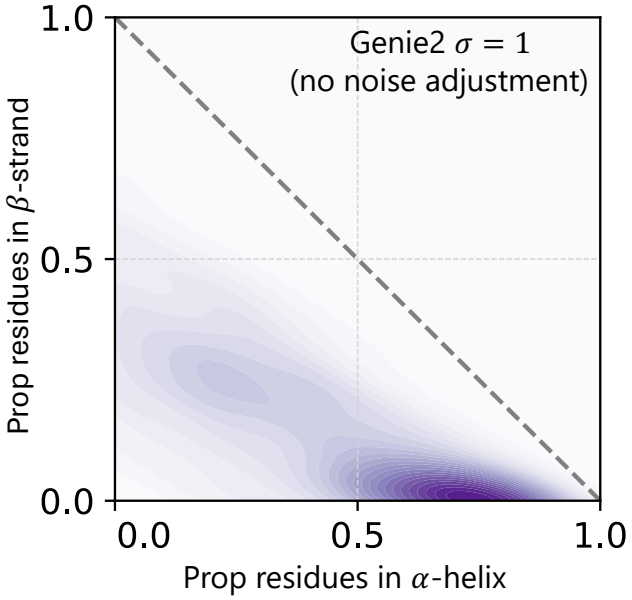

Figure 3. Contour plot of the joint distribution of alpha-helical and beta-strand composition for the base Genie2 model (i.e. without scaling the noise).

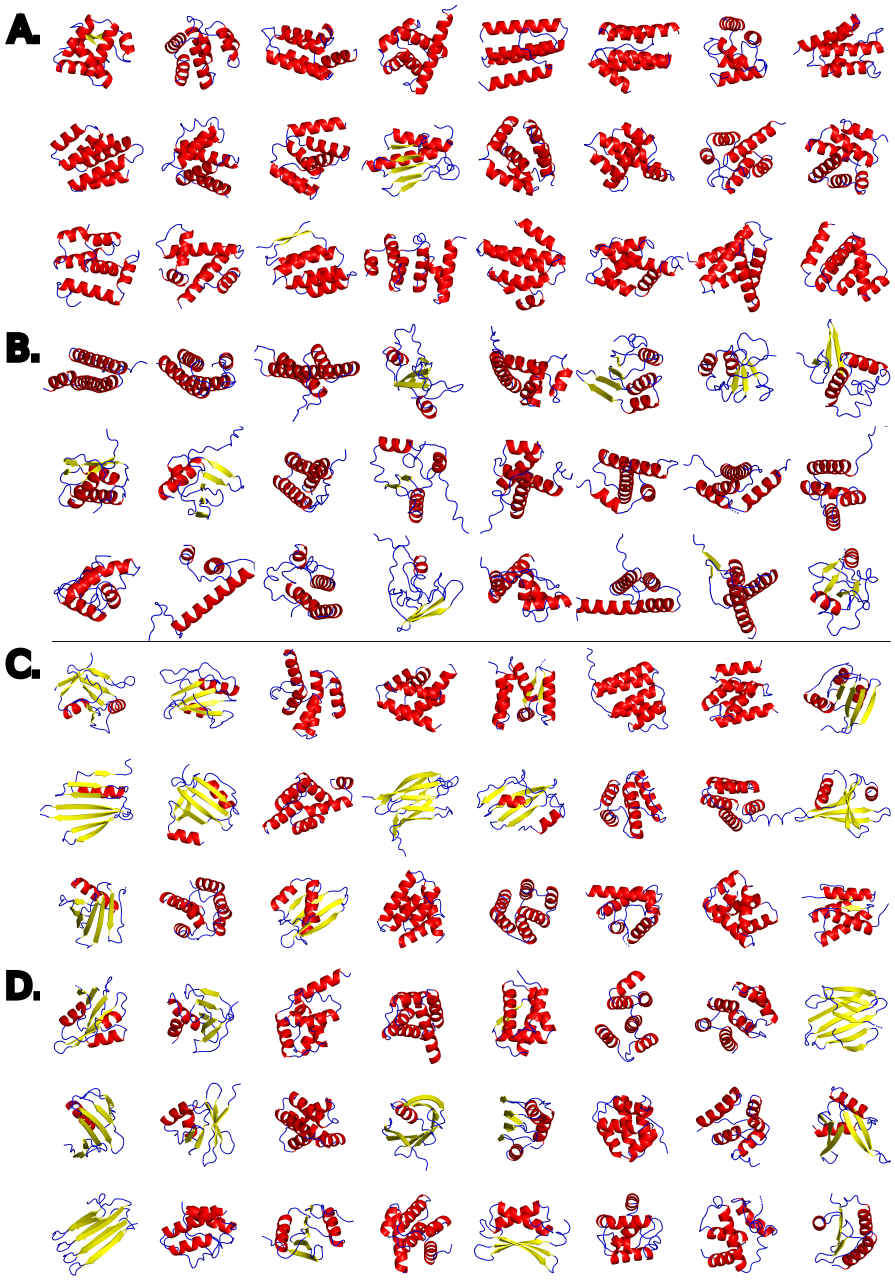

Figure 4. Protein backbones colored by secondary structure type sampled from **A**: Genie2 base model with reduced noise scale $\sigma = 0.5$. **B**: Genie2 base model with original noise scale $\sigma = 1.0$. **C**: Genie2 model fine-tuned by CGM-relax with $N = 9$ univariate constraints and $\lambda = 10^2/(2N) \approx 5$. **D**: Genie2 model fine-tuned by CGM-relax with $N = 9$ univariate constraints and $\lambda = 10^4/(2N) \approx 500$. Structures are visualized with pymol. To allow pymol annotation of secondary structure, N, C, and O backbone atoms are inferred from the C only Genie2 samples with CG2AA (Lombardi et al., 2016).