Proceedings of Machine Learning Research vol 284:1–37, 2026 20th Conference on Neurosymbolic Learning and Reasoning

Learning Deep Generative Models under Hard Linear Equality Constraints

Ruoyan Li LIRUOYAN2002@G.UCLA.EDU
Dipti Ranjan Sahu DIPTISAHU11@G.UCLA.EDU
Guy Van den Broeck GUYVDB@CS.UCLA.EDU
University of California, Los Angeles
Zhe Zeng ZHEZ@VIRGINIA.EDU
University of Virginia

Editors: Alessandra Mileo, Andrea Passerini and Cogan Shimizu

Abstract

While deep generative models (DGMs) have demonstrated remarkable success in capturing complex data distributions, they consistently fail to learn strict symbolic constraints that encode domain knowledge, thus requiring a neurosymbolic approach to constraint integration. Existing solutions to this challenge have primarily relied on projection-based methods and often ignore the underlying data distribution, harming generative performance. In this work, we propose a probabilistically sound neurosymbolic framework for enforcing hard linear equality constraints into DGMs to generate constraint-compliant and realistic data. This is achieved by our proposed gradient estimators that allow the constrained distribution, the data distribution conditioned on constraints, to be differentiably learned. We carry out extensive experiments with various DGM model architectures over five image datasets and three scientific applications in which domain knowledge is governed by linear equality constraints. Among all the constraint integration strategies, ours not only guarantees the satisfaction of constraints in generation but also archives superior generative performance across every benchmark.

1. Introduction

Deep generative models (DGMs) have made great progress in generating realistic data by capturing the underlying patterns and distributions of a dataset. However, they have been found to struggle with implicitly learning strict, symbolic domain knowledge (Zhang et al., 2023). For example, if a chemist trains a model on a charge-neutral molecule dataset for predicting charges of each atom in a given molecule, they want the predicted charges to sum up to zero, satisfying the charge neutrality property as explicit symbolic background knowledge. This is a true concern for chemists in Raza et al. (2020) as they find state-of-the-art models almost surely generate predictions that violate this property and thus are useless for downstream tasks. The ability to incorporate symbolic domain knowledge into generative modeling remains crucial for broad application of AI.

Symbolic domain knowledge such as the one shown above takes the form of *linear-equality constraints*, an important class of constraints that have been studied by many given their wide applications. Another example is the mass balance and stoichiometry in chemical engineering whose processes are governed by linear equality constraints (Chen et al., 2024). Such constraints are also necessary for stock investment allocation in financial engineering (Zhang et al., 2020; Butler and Kwon, 2021). While this list can be made longer, it is surprisingly challenging to enforce these seemingly simple constraints into DGMs.

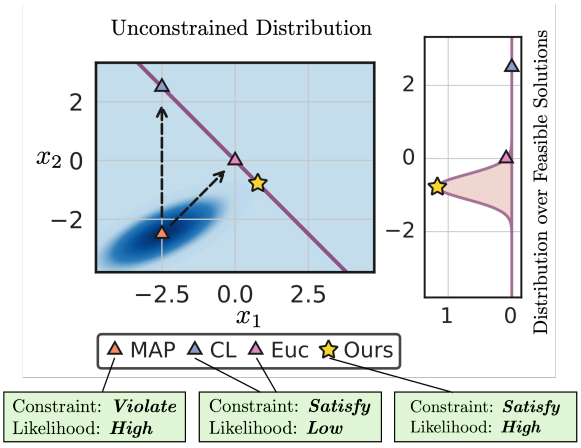

Figure 1: Comparison of different methods for generating samples that satisfy linear equality constraints. The left panel shows the original unconstrained distribution in a 2-dimensional plane, with the purple line representing the constraint $x_1 + x_2 = 0$. Our proposed method generates the most realistic sample as indicated by the right figure, outperforming existing methods that optimize for L1 distance (CL) and L2 distance (Euc).

Incorporating symbolic constraints directly into the differentiable learning process can improve both generalization and data efficiency, compared to imposing them only in post-processing steps (Jeon et al., 2025). Existing methods for enforcing such constraints more or less share the same philosophy: first generating a candidate prediction by sampling from the unconstrained distribution, which is almost certainly to violate the constraints, and then making *minimal changes* to the candidate such that it satisfies the constraint and is returned as the final prediction. They differ in the *heuristics* proposed for deriving minimal changes and accordingly the *gradient estimators* for differentiating through the samples. For example, given a candidate sample $\boldsymbol{x}$, Stoian et al. (2024) propose a *Constraint Layer* (CL) that incrementally updates i-th feature $\boldsymbol{x}_i$ to return a sample $\tilde{\boldsymbol{x}}$ that satisfies the constraints while Chen et al. (2024) (Euc) propose to use $L2$ distance and formulate such projection as quadratic programming problems. We visualize these baselines works at deployment in Figure 1 where the arrows indicate the minimal changes to a given maximum a posteriori (MAP) solution as a candidate. We observe that while these predictions are obtained by making minimal perturbations to the MAP solution, they result in low-likelihood constrained samples. This is because these transformations only consider sample distances but disregard the underlying data distribution learned by DGMs.

In this work, we present a probabilistically sound framework to incorporate the hard linear equality constraints into the DGMs based on a simple yet effective idea: instead of *transforming the sample*, we propose to *transform the distribution*. While the baseline methods consider first sampling and then constrain the samples, we propose to first constrain the distribution to the feasible space satisfying the constraints, and then sample from the constrained distribution. For example, in Figure 1, we generate our sample by first deriving the distribution on the right figure and then taking the MAP of the constrained distribution. The benefits of this approach are two-fold: the resulting samples are guaranteed to satisfy the constraints and, they closely resemble the true data with high likelihood.

Specifically, we make the following **contributions**. (i) We propose and compare several gradient estimators that allow the end-to-end training of such constrained distributions. We further validate that one specific design of gradient estimator is significantly more effective than the others. (ii) We demonstrate that our approach is flexible in two key ways: it is agnostic to the DGM architectures, making it applicable across a wide range of models; also, it allows constraints to be seamlessly integrated into any layer of the DGM. (iii) To conduct

extensive empirical evaluations, we compare the different constraint enforcement approaches across five image datasets and three scientific applications, involving multiple DGM variants. (iv) We show that standard DGMs always fail to learn the constraint, underscoring the need for explicit constraint enforcement to generate realistic samples compliant with the domain knowledge. (v) We further demonstrate that our approach consistently achieves better generative performance than all baseline methods across every benchmark. *Overall, our approach paves the way for a principled design of integrating constraints into DGMs, enabling the generation of constraint-compliant, realistic samples.*

2. Related Work

Our work lies in the field of neuro-symbolic AI where integrating constraints as background knowledge into deep learning models is widely studied (Garcez and Lamb, 2023).

Constraint Enforcement. There are multiple ways to enforce constraints in the neural networks. Some directly incorporate background knowledge as network layers (Ahmed et al., 2022; Giunchiglia and Lukasiewicz, 2021) while some others propose new architectures (Flores et al., 2025). In the context of generative tasks, Di Liello et al. (2020) embed propositional logic constraints into Generative Adversarial Networks (GANs) for structured object generation, while Misino et al. (2022) integrate probabilistic logic programming (De Raedt et al., 2007) with Variational Autoencoders (VAEs). Hendriks et al. (2020) impose linear operator constraints within the architecture of feedforward neural networks, but their approach does not generalize to other model architectures. Wang et al. (2023) enforce positive linear constraints using a Sinkhorn algorithm, where their method is only applicable to output variables being unit hypercubes. Chen et al. (2024) formulate constraint satisfaction as an optimization problem. They use the $L2$ distance and derive projections based on the Karush–Kuhn–Tucker (KKT) conditions. Amos and Kolter (2017) and Donti et al. (2017) integrate quadratic programming solvers as differentiable modules within end-to-end trainable deep networks while some other works formulate it as submodular optimization problems Djolonga and Krause (2017), Tschiatschek et al. (2018), and Wilder (2019). Most of these approaches ignore underlying data distributions when solving optimization problems while in our method we leverage the information from the constrained distribution for optimizing the DGMs. Quantitative comparisons between our method and existing work are further presented in the experimental section.

Constrained Sampling. There are some existing works in the field of statistical modeling that study how to sample from linearly constrained Gaussian distributions that can be potentially applied as post-processing steps. For example, Vrins (2018) investigates a linear weighted constraint for independent standard Gaussian variables, while Lamboni (2022) focuses on a fixed-sum constraints for independent Gaussian variables with zero means. Jeon et al. (2025) proposes projection-free stochastic dynamics for sampling from constrained distributions with guarantees on the convergence. These methods are not differentiable and thus not applicable to DGM training. Similarly to ours, Narasimhan et al. (2025) considers constrained sampling during the denoising process for diffusion models, but it is based on projections and thus requires a large number of denoising steps to nudge the sample generation towards the constrained set.

Exactly-k Constraints. The discrete counterpart of linear equality constraints, the exactly-k constraints defined as $\sum_i x_i = k$ with x_i being categorical variables is studied by many. Maddison et al. (2017) and Jang et al. (2017) propose similar ideas to refactor the non-differentiable sample from a categorical distribution with a differentiable sample from Gumbel-Softmax distributions. Other gradient estimators for this constraint either employ variants of score function and straight-through estimator or propose certain relaxations (Kim et al., 2016; Chen et al., 2018; Grover et al., 2019; Xie and Ermon, 2019). Closely related to our work is a recently introduced gradient estimator (Ahmed et al., 2023) that leverages the constrained marginal distribution as a proxy for differentiation.

Soft Constraints. A line of research integrates the constraints by optimizing for the probability of constraint satisfaction, encouraging the model to generate compliant samples (Diligenti et al., 2012; Xu et al., 2018; Fischer et al., 2019; Badreddine et al., 2022; Stoian et al., 2023; Shukla et al., 2024). This is achieved by modifying the loss function with differentiable constraint probabilities. Barthakur and Chamon (2025) optimize the loss function under a constrained space where the constraints are expected to be satisfied. However, these methods do not guarantee the satisfaction of the constraint.

3. Problem Statement

Instead of sampling from unconstrained DGM $p_{\boldsymbol{\theta}}(\boldsymbol{z})$, our goal is to constrain it as below

$$\boldsymbol{\theta} = h_{\boldsymbol{v}}(\boldsymbol{x}), \quad \boldsymbol{z} \sim p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k}), \quad \hat{\boldsymbol{y}} = f_{\boldsymbol{u}}(\boldsymbol{z}), \tag{1}$$

where $\boldsymbol{x} \in \mathcal{X}$ and $\hat{\boldsymbol{y}} \in \mathcal{Y}$ denote feature inputs and target outputs, respectively, $h_{\boldsymbol{v}} : \mathcal{X} \to \Theta$ and $f_{\boldsymbol{u}} : \mathcal{Z} \to \mathcal{Y}$ are smooth, parameterized mappings. Parameters $\boldsymbol{\theta}$ induce a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ over the latent variables $\boldsymbol{z}$ where parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ consist of the mean vector $\boldsymbol{\mu} \in \mathbb{R}^n$ and the covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$. That is, $\boldsymbol{z}$ has its probability density function (p.d.f.) defined as $p_{\boldsymbol{\theta}}(\boldsymbol{z}) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}_{\boldsymbol{\theta}}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\theta}})^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1}(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\theta}})\right)$. $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{k}$ denotes the linear equality constraints with $\boldsymbol{A} \in \mathbb{R}^{a \times n}$, $\operatorname{rank}(\boldsymbol{A}) = a \leq n$, and $\boldsymbol{k} \in \mathbb{R}^a$, enforced over the DGM $p_{\boldsymbol{\theta}}(\boldsymbol{z})$ inducing a constrained distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})$.

This formulation is general and it subsumes various DGM classes that integrate the linear equality constraint in *1) latent space*, where the model learns a constrained posterior distribution $p_{\boldsymbol{\theta}}$ over latent variables.; or *2) output* (when $f_{\boldsymbol{u}}$ is the identity function), where the model learns the parameters $\boldsymbol{\theta}$ to approximate the underlying data distribution. When constraints are located in the latent space, we introduce gradient estimators to differentiate through the constrained distribution in Section 4. When constraints are enforced in the output space, we present closed-form expected loss functions for standard auto-differentiation in Section 4.3. The training of this model is by optimizing an expected loss:

$$L(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\omega}) = \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})}[\ell(f_{\boldsymbol{u}}(\boldsymbol{z}), \boldsymbol{y})] \quad \textit{with} \quad \boldsymbol{\omega} = (\boldsymbol{v}, \boldsymbol{u}) \quad \textit{and} \quad \boldsymbol{\theta} = h_{\boldsymbol{v}}(\boldsymbol{x}), \tag{2}$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ is a point-wise loss function.

4. Gradient Estimation for Linear Equality

Standard auto-differentiation can not be directly applied to the expected loss due to two main obstacles. First, for the gradient of the expected loss L w.r.t. parameters $\boldsymbol{u}$ in the

Table 1: Summary of gradient estimators. The first block presents baseline estimators and the second block presents our proposed ones.

Gradient Estimator	**Proxy $m(\boldsymbol{\theta})$**	**Description**
Random	–	Sample a random gradient from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$.
Unconstrained Marginal	$p_{\boldsymbol{\theta}}(z_i)$	p.d.f. of unconstrained $\boldsymbol{z}$ as a proxy for $\boldsymbol{z}$.
Constrained Layer	$\mathrm{CL}\big(\boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}\big)$	Use reparametrization trick as a proxy for $\boldsymbol{z}$. Constrained Layer enforces $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{k}$.
Constrained Reparametrization	$\boldsymbol{z_1} = \overline{\boldsymbol{\mu}} + \overline{\boldsymbol{\Sigma}}\boldsymbol{\epsilon}$ $\boldsymbol{A}[\boldsymbol{z_1}, \boldsymbol{z_2}] = \boldsymbol{k}$	Apply reparametrization to constrained distribution. Solve the linear system to get remaining values.
Constrained Marginal	$p_{\boldsymbol{\theta}}\big(z_i \mid \boldsymbol{A}\,\boldsymbol{z} = \boldsymbol{k}\big)$	p.d.f. of conditional marginals as a proxy for $\boldsymbol{z}$.
Marginal Expectation	$\mathbb{E}_{z_i \sim p_{\boldsymbol{\theta}}(\,z_i \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})}[\,z_i\,]$	Expectation of conditional marginals as a proxy for $\boldsymbol{z}$.

decoder mapping $f_{\boldsymbol{u}}$,

$$\nabla_{\boldsymbol{u}} L(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\omega}) = \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})} \partial_{\boldsymbol{u}} f_{\boldsymbol{u}}(\boldsymbol{z}, \boldsymbol{x})^{\top} \nabla_{\hat{\boldsymbol{y}}} \ell(\hat{\boldsymbol{y}}, \boldsymbol{y}) \tag{3}$$

with $\hat{y} = f_{\boldsymbol{u}}(\boldsymbol{z})$ being the decoding of a latent sample $\boldsymbol{z}$, this expectation does not allow closed-form solution in general and requires Monte-Carlo estimations by sampling $\boldsymbol{z}$ from the constrained distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})$. Another issue arises in the gradient of L w.r.t. parameters $\boldsymbol{v}$ in the encoder mapping which is defined as

$$\nabla_{\boldsymbol{v}} L(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\omega}) = \partial_{\boldsymbol{v}} h_{\boldsymbol{v}}(\boldsymbol{x})^{\top} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\omega}). \tag{4}$$

The obstacle lies in the computation of the gradient of the expected loss L w.r.t. $\boldsymbol{\theta}$ defined as $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\omega}) := \nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})}[\ell(f_{\boldsymbol{u}}(\boldsymbol{z}, \boldsymbol{x}), \hat{\boldsymbol{y}})]$ which requires gradient estimators. In this section, we tackle the gradient estimation for the linear equality constraint by solving the aforementioned two subproblems: **(P1)** how to *sample exactly* from the constrained distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})$ and **(P2)** how to *estimate* $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\omega})$. For **(P1)**, we observe that the constrained distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})$ is a multivariate Gaussian distribution and thus performing exact sampling is straightforward as long as we derive the parameters for the constrained distribution. We formally state this observation in Appendix J.1. In the following, we present the various design choices as candidate solutions to **(P2)**.

4.1. Gradient Estimator Design

We propose novel ways to build gradient estimators that are able to leverage the constraint information. We first propose an approximation to the problematic term in Equation 4 as

$$\nabla_{\boldsymbol{\theta}} L(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\omega}) \approx \partial_{\boldsymbol{\theta}} \boldsymbol{m}(\boldsymbol{\theta}) \nabla_{\boldsymbol{z}} \ell(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\omega}), \tag{5}$$

where $\boldsymbol{m}(\boldsymbol{\theta})$ should be chosen as a function that can be efficiently computed and differentiated and meanwhile encode constraint information. Here, we consider two candidates for $\boldsymbol{m}(\boldsymbol{\theta})$: 1) the conditional marginal probability density $p_{\boldsymbol{\theta}}(z_i | \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})$; and 2) the expectation of z_i under the conditional marginal, that is, $\mathbb{E}_{z_i \sim p_{\boldsymbol{\theta}}(z_i | \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})}[z_i]$. The intuition behind the adoption of these marginal distributions is that, by conditioning on the constraints, their gradients provide a differentiable proxy for optimizing the constrained distribution. It

Figure 2: Comparison of gradient estimators when point-wise loss is L1 (left) or L2 (right). y-axis shows the cosine distance between estimated vs. ground-truth gradient directions.

encourages the constrained model to generate constraint-compliant samples with low loss, allowing for efficient end-to-end training of DGMs.

While these two quantities seem to encode similar information, we make an interesting observation in our empirical study that in continuous domains, the use of expectation is consistently more effective than conditional marginals. We further provide a baseline estimator that chooses $\boldsymbol{m}(\boldsymbol{\theta})$ to be the unconstrained marginals $p_{\boldsymbol{\theta}}(z_i)$, meaning that the constraint is ignored during the training process.

The remaining question is how to compute and differentiate $\boldsymbol{m}(\boldsymbol{\theta})$ for these two different estimators. We present below the theoretical results to show that constrained marginals and their expectations admit closed-form representation and thus allow efficient computations.

Proposition 1 (Gaussian Conditional Marginal and Expectations) *Given* $\boldsymbol{z} = (z_1, \ldots, z_n)^T \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$*, the conditional marginal* $p_{\boldsymbol{\theta}}(z_i \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})$ *follows a univariate Gaussian distribution with mean* $\overline{\mu}_i = \mu_i + \boldsymbol{e}_i^T \boldsymbol{\Sigma} \boldsymbol{A} \left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1} (\boldsymbol{k} - \boldsymbol{A}\boldsymbol{\mu})$ *and variance* $\overline{\sigma}_i^2 = \boldsymbol{e}_i^T \boldsymbol{\Sigma} \boldsymbol{e}_i - \boldsymbol{e}_i^T \boldsymbol{\Sigma} \boldsymbol{A}^T \left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1} \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{e}_i$*. Further, the expectation of the marginal is* $\overline{\mu}_i$*.*

The reparameterization trick (Kingma and Welling, 2014) is perhaps the most commonly used technique for differentiating through random samples. Specifically, it expresses a sample $\boldsymbol{z}$ as $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim N(\boldsymbol{0}, \boldsymbol{I})$ and $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are mean and standard deviation, respectively. Since the constrained distribution is also Gaussian, we also propose to apply reparametrization trick to the constrained distribution and solve the linear system to acquire the remaining variables. In the following experiment sections, we show that while the *Constrained Reparametrization* is a competing baseline, it consistently performs worse than *Marginal Expectation* under all settings.

4.2. Comparison of Gradient Estimators

We present a rigorous comparison of all aforementioned gradient estimator designs summarized in Table 1: we consider a synthetic setting where the ground truth gradients can be obtained by taking derivatives of a closed-form expected loss which will be described in Section 4.3 such that we can compare the performance of gradient estimations. The distance between the estimated and the ground truth gradient vectors is measured by the cosine distance, a commonly used metric for evaluating the gradient estimators (Ahmed et al., 2023; Shekhovtsov et al., 2020; Ilyas et al., 2020; Li et al., 2021). We evaluate the performance of gradient estimators on three metrics: bias, variance, and average error.

We further include three baseline estimators, *Random*, *Unconstrained Marginal*, *Constrained Layer* as defined in Table 1. While the Constrained Layer introduced by Stoian

et al. (2024) cannot be directly utilized as a gradient estimator, we utilize the reparameterization trick to facilitate backpropagation through the random sampling process and constrained layer to enforce equality constraints.

Results are shown in Figure 2, where *Marginal Expectation* significantly outperforms the others in all cases. *Unconstrained Marginal* has similar performances to *Random* which is expected since it discards the constraint information. What is interesting is that *Constrained Marginal*, even though it is informed by constraint, it also performs as bad as *Random*. In Bernoulli setting, *Marginal Expectation* and *Constrained Marginal* are the same estimator as shown in Ahmed et al. (2023) while we show that in the Gaussian setting, the former is capable of providing decent gradient approximations while the latter is not. We refer the readers to Appendix C for additional experimental details.

We provide additional experiment results in Appendix N by varying the the dimension n. While *Constrained Reparametrization* is an unbiased estimator, it suffers from high variance in the finite-sample setting. It might be hard to see in Figure 2, but as is clear from Figure 11, the variance of *Constrained Reparametrization* is an order of magnitude worse than *Marginal Expectation*, which explains why *Constrained Reparametrization* gives large errors in its estimations in most settings except when the dimension is low with $n = 10$.

Note that the metrics are defined with cosine similarity and not to be confused with those in the classic Monte Carlo estimation setting. The cosine similarity renders the bias to be more sensitive to the bad samples: for example, if there's one sample with its norm way greater than the others, it might dominate the direction of the estimated gradient and lead to high error in cosine similarity. That is, while *Constrained Reparametrization* is an unbiased estimator for the gradient vector with bias defined in the classic setting, it can be biased in estimating the angle/direction with bias defined with cosine similarity for measuring directions. In contrast, our estimator *Marginal Expectation* remains robust and delivers more accurate gradients in high-dimensional regimes.

4.3. Closed-Form Expected Loss

We further explore the special case when gradient estimators are not necessary. It holds when the expected loss in Equation 2 admits closed-form expressions, allowing standard training to be applied and thus no gradient estimation is needed. For such cases to hold, the first assumption we make is that the mapping $f_{\boldsymbol{u}}$ is an identity function, that is, $\hat{\boldsymbol{y}} = \boldsymbol{z}$, as it can introduce high non-linearity. Then we show that when the element-wise loss ℓ is the L1 or L2 loss, the expected loss admits a closed-form expression as below.

Proposition 2 (Gaussian Closed-form Expected Loss) *Let $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Let $\boldsymbol{y} = (y_1, \ldots, y_n)^T$ be the ground truth vector subject to the equality constraint $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{k}$. Then it holds that i) when ℓ is L1 loss, $L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \overline{\boldsymbol{\Sigma}}_{i,i} \sqrt{\frac{2}{\pi}} e^{-\frac{(\overline{\boldsymbol{\mu}}_i - y_i)^2}{2\overline{\boldsymbol{\Sigma}}_{i,i}^2}} + (\overline{\boldsymbol{\mu}}_i - y_i)\, erf\left(\frac{\overline{\boldsymbol{\mu}}_i - y_i}{\sqrt{2}\,\overline{\boldsymbol{\Sigma}}_{i,i}}\right)$; when ℓ is L2 loss, $L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \overline{\boldsymbol{\mu}}_i^2 + \overline{\boldsymbol{\Sigma}}_{i,i}^2 - 2y_i\overline{\boldsymbol{\mu}}_i + y_i^2$, where $\overline{\boldsymbol{\mu}}$ and $\overline{\boldsymbol{\Sigma}}$ are defined above.*

5. Experiments

5.1. VAE with Constrained Latent Space

We consider an experiment setup where the VAE model has its latent space constrained by linear equality as regularization. The VAE is trained on the MNIST dataset using the

evidence lower bound (ELBO) as objective, which consists of a reconstruction loss (RL) and the KL divergence between a constrained approximate posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k}, \boldsymbol{x})$ and a prior of the latent space. The generative performance is evaluated using test negative likelihood, estimated using importance sampling (Burda et al., 2016), negative ELBO, and reconstruction loss. Experiment results are presented in Figure 3 where the estimator *Marginal Expectation* outperforms the other estimators in all three metrics, consistent with synthetic experimental results in Figure 2.

5.2. Constrained Generation using VAE

We modify MNIST dataset by standardizing overall brightness of each image using a linear equality constraint. Three VAE models are considered: Vanilla VAE (Kingma and Welling, 2014), Ladder VAE (Sønderby et al., 2016), and Graph VAE (He et al., 2018). We compare the performance of these models and their constrained counterparts integrated with linear equality using estimator *Marginal Expectation*. We refer the readers to Appendix E for model implementations and data modification. We measure a relaxed constraint violation rate, which calculates the proportion of reconstructed samples that violate constraints within some tolerance.

Table 2: Comparison on VAE generative performance. Our constrained models achieve similar or better generative ability while strictly satisfying the constraints.

Model	LL ↑	ELBO ↑	RL ↓	Violation ↓
VAE	-22.42 ± 0.29	-23.41 ± 0.22	15.00 ± 0.46	0.30 ± 0.06
VAE + Constr Layer	-34.45 ± 2.64	-40.89 ± 9.37	37.11 ± 9.35	**0.00 ± 0.00**
VAE + Constr Reparam	-22.33 ± 0.48	-23.83 ± 0.49	14.54 ± 0.58	**0.00 ± 0.00**
VAE + Mar Exp (ours)	**-21.48 ± 0.18**	**-22.62 ± 0.07**	**12.79 ± 0.11**	**0.00 ± 0.00**
Ladder VAE	-24.25 ± 0.07	-30.84 ± 0.51	**23.06 ± 0.54**	0.38 ± 0.02
Ladder VAE + Constr Layer	-36.83 ± 0.56	-39.59 ± 0.56	37.46 ± 0.55	**0.00 ± 0.00**
Ladder VAE + Constr Reparam	-25.27 ± 0.19	-31.56 ± 0.48	25.20 ± 0.62	**0.00 ± 0.00**
Ladder VAE + Mar Exp (ours)	**-23.86 ± 0.06**	**-30.78 ± 0.08**	**23.40 ± 0.16**	**0.00 ± 0.00**
Graph VAE	-22.74 ± 0.11	-23.54 ± 0.18	15.45 ± 0.41	0.29 ± 0.09
Graph VAE + Constr Layer	-33.27 ± 3.60	-33.27 ± 5.84	28.29 ± 6.40	**0.00 ± 0.00**
Graph VAE + Constr Reparam	-22.96 ± 0.80	-23.55 ± 0.86	16.08 ± 1.38	**0.00 ± 0.00**
Graph VAE + Mar Exp (Ours)	**-21.61 ± 0.20**	**-22.53 ± 0.06**	**12.73 ± 0.21**	**0.00 ± 0.00**

Reults. As shown in Table 2, we find out that, even under the relaxed metric, the unconstrained VAEs have high constraint violation rates. On the contrary, our method can constrain the model such that their generated data satisfy the constraint while also achieving better generative performance. Although *Constrained Layer* can precisely enforce the constraint, it significantly diminishes the generative capability. We also observe that *Marginal Expectation* adds negligible computational overhead.

5.3. Constrained Generation using Diffusion Models

We consider diffusion models (Ho et al., 2020; Song et al., 2021; Song and Ermon, 2019). We show that our method can be integrated into backward diffusion process and not only improve sample quality but also ensure constraint satisfaction. We modify CIFAR 10 (Krizhevsky, 2009), CelebA (Liu et al., 2015), LSUN Church, and LSUN Cat (Yu et al., 2015) datasets by standardizing the overall brightness using linear equality constraints. We refer to Appendix F for modification of datasets and Figure 5 for inference pipeline.

DDPM. The model follows the standard DDPM (Ho et al., 2020). During training, Gaussian noise is incrementally added, and an iterative denoising process is performed to progressively generate samples from Gaussian noise. We incorporate our exact sampling methods into the backward diffusion process. Instead of predicting $p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)$, we generate $p_\theta(\boldsymbol{x}_0 \mid \boldsymbol{x}_1, \boldsymbol{A}\boldsymbol{x}_0 = \boldsymbol{k})$. The results are presented in Table 3.

Table 3: Comparison of constrained and unconstrained models across datasets under DDPM. We report FID (Fréchet Inception Distance), IS (Inception Score), and Violation metrics.

Dataset	Model	FID ↓	IS ↑	Violation ↓
CIFAR	Ours	**3.811**	9.223 ± 0.130	**0**
	Constr Layer	4.234	8.535 ± 0.157	**0**
	Constr Reparam	3.881	9.212 ± 0.122	**0**
	DDPM	4.173	**9.278 ± 0.116**	0.999
CelebA	Ours	**10.193**	**2.360 ± 0.016**	**0**
	Constr Layer	12.067	2.345 ± 0.039	**0**
	Constr Reparam	10.961	2.043 ± 0.028	**0**
	DDPM	10.345	2.358 ± 0.030	0.999
LSUN Church	Ours	**4.779**	**2.471 ± 0.020**	**0**
	Constr Layer	6.695	2.319 ± 0.028	**0**
	Constr Reparam	4.895	2.377 ± 0.032	**0**
	DDPM	4.945	2.460 ± 0.028	1.0
LSUN Cat	Ours	**12.489**	**4.711 ± 0.054**	**0**
	Constr Layer	13.472	4.642 ± 0.081	**0**
	Constr Reparam	12.696	4.707 ± 0.062	**0**
	DDPM	12.913	4.705 ± 0.047	1.0

Table 4: Performances of different methods for estimating partial charges on metal ions.

Method neutrality enforcement	**MAD ↓** mean ± std	**NLL ↓** mean ± std
Constrained Layer	0.327 ± 0.004	103.522 ± 3.018
Constant Prediction	0.324 ± 0.007	—
Element-mean (uniform)	0.154 ± 0.002	—
Element-mean (variance)	0.153 ± 0.002	—
MPNN (KKThPINN)	0.0260 ± 0.0008	109.8 ± 6.9
MPNN (Constr Reparam)	0.0256 ± 0.0007	334.03 ± 2398
MPNN (variance)	0.0251 ± 0.0010	-19.9 ± 71.1
Closed-form (ours)	**0.0245±0.0009**	> 1e+7
Likelihood (ours)	0.0248 ± 0.0008	**-252±24.7**
Constrained Layer (ens)	0.319 ± 0.002	99.236 ± 2.3
MPNN (ens, KKThPINN)	0.0244 ± 0.0006	57.29 ± 12.8
MPNN (ens, Constr Reparam)	0.0242 ± 0.0006	4883.46 ± 1695
MPNN (ens, variance)	0.0238 ± 0.0007	-45.2 ± 55.8
Closed-form (ens, ours)	**0.0230±0.0008**	> 1e+7
Likelihood (ens, ours)	**0.0231±0.0007**	**-180±38.3**

DDIM. Inspired by recent success in incorporating guidance at intermediate backward diffusion steps (Yuan et al., 2023; Liu et al., 2024), we also incorporate our method in selected backward diffusion steps under DDIM sampling mechanism. Using the CIFAR 10 dataset, we explore what would be the optimal schedule policy and the optimal number of constrained sampling steps. We refer the readers to Appendix F for additional details. We use this scheduling policy on all constrained models and compare with unconstrained counterparts in Table 8.

Results. Our experiment results demonstrate that standard diffusion models rarely satisfy the imposed constraints, despite being trained on data distribution that is governed by these constraints. In contrast, diffusion models incorporating our method have guaranteed constraint satisfaction. Moreover, they exhibit superior generative performance.

5.4. Charge-Neutral Predictions

Predicting partial charges in metal-organic frameworks (MOFs) is essential but challenging due to the strict charge neutrality constraint. Following Raza et al. (2020), we represent MOF structures as undirected graphs $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ with adjacency matrix $\mathbf{D}$. We use a Message Passing Neural Network (MPNN) to predict a Gaussian charge distribution, $\mathbf{q} = r(\mathbf{X}, \mathbf{D})$, subject to the neutrality condition $\sum_{v=1}^{n} q_v = 0$.

To improve upon standard L1 loss training, we introduce a closed-form Gaussian loss and the negative log-likelihood (NLL) of a constrained multivariate Gaussian. We additionally utilize a two-model ensemble method with varied initializations to improve robustness. As shown in Table 4, our NLL and closed-form loss approaches outperform the strongest baseline. Conversely, strictly enforced methods like *Constrained Layer* maintain neutrality but severely impair predictive accuracy.

5.5. Chemical Process Units and Subsystems

Linear equality constraints are essential in chemical engineering, governing processes through principles like mass balance and stoichiometry (Chen et al., 2024). We use machine learning

Table 5: The mean and standard deviation of MSE scaled by 10^{-4} are reported. All experiments are averaged over 10 runs.

Model	CSTR	plant	distillation
ECNN	20.6±27.0	0.31±0.23	1.94±0.70
KKThPINN	11.7±20.3	0.11±0.04	2.02±0.94
NN	18.3±20.8	0.34±0.64	1.99±0.67
PINN	260.8±20.4	3.62±1.94	40.9±10.7
CL	9.28±3.56	0.58±0.64	2.26±1.19
Constr Reparam	7.13±3.22	0.14±0.07	2.22±0.94
Ours	**4.31±1.58**	**0.09±0.05**	**1.73±0.70**

Table 6: Comparison of models based on Sharpe ratio. We report the mean and standard deviation averaged over 10 runs.

Model	Sharpe ratio ↑
StemGNN	1.5576±0.3405
StemGNN-KKThPINN	1.8092±0.7055
StemGNN-CL	1.5018±0.3318
StemGNN-Constr Reparam	1.8162±0.2966
Ours	**1.9041±0.2329**

surrogate models to provide physically accurate representations of these systems. We follow the experiment setting from Chen et al. (2024) and conduct experiments on aspen models of a continuous stirred-tank reactor (**CSTR**) unit, an extractive distillation subsystem (**distillation**), and a chemical plant (**plant**).

We enforce a Gaussian distribution assumption to the target variables. Leveraging our theoretical framework, we sample exactly from the constrained distribution and train the model using closed-form expected loss functions. We compare the performance of our approach to several baselines ECNN (Chen et al., 2024), *KKThPINN* (Chen et al., 2024), standard Neural Networks (NN), Physics Informed Neural Networks (Raissi et al., 2019), and *Constrained Layer* (Stoian et al., 2024). We refer the readers to the Appendix H and Chen et al. (2024) for detailed information regarding baseline implementations and experiment settings. As shown in Table 5, our model consistently outperforms the baselines by a significant margin. Furthermore, as detailed in Appendix H, our method also achieves significantly faster convergence.

5.6. Stock Investment

Stock investment allocation (Zhang et al., 2020; Butler and Kwon, 2021) optimizes portfolio distribution based on predicted market trends to maximize returns under uncertainty. A strict requirement of this process is that all assigned stock weights must sum to exactly 1. Using one year of historical S&P 500 data, our objective is to construct a portfolio that maximizes the Sharpe ratio (Sharpe, 1966) over the subsequent 120 trading days.

We employ StemGNN (Cao et al., 2020), a state-of-the-art time series prediction model, utilizing a default softmax activation to enforce the sum-to-one constraint. By permitting short selling, we assume the allocation weights follow Gaussian distributions. Model performance is evaluated using the Sharpe ratio, defined as Sharpe Ratio $= \frac{R_p - R_f}{\sigma_p}$, where R_p is the expected return, R_f is the risk-free rate, and σ_p is the standard deviation of excess return. Results in Table 6 shows that our model outperforms the baseline alternatives.

6. Conclusion

We introduced a principled framework for incorporating hard linear equality constraints into DGMs. Our approach directly constrains the distribution and enables end-to-end training of the constrained models through novel gradient estimators and closed-form expected loss.

Acknowledgments

The authors acknowledge the Research Computing at the University of Virginia. This work was funded in part by the DARPA ANSR and CODORD programs under awards FA8750-23-2-0004 and HR00112590089, and gifts from Cisco Research, Qualcomm, and Amazon.

References

Kareem Ahmed, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck, and Antonio Vergari. Semantic probabilistic layers for neuro-symbolic learning. *Advances in Neural Information Processing Systems*, 35:29944–29959, 2022.

Kareem Ahmed, Zhe Zeng, Mathias Niepert, and Guy Van den Broeck. Simple: A gradient estimator for k-subset sampling. In *Proceedings of the International Conference on Learning Representations (ICLR)*, may 2023.

Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 136–145. PMLR, 2017.

Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artificial Intelligence*, 303:103649, 2022.

Aneesh Barthakur and Luiz FO Chamon. Learning with statistical equality constraints. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations (ICLR)*, 2016.

Alex Butler and Ryan Kwon. Integrating prediction in mean-variance portfolio optimization. *Available at SSRN 3788875*, 2021.

Daoyuan Cao, Yue Wang, Jinjing Duan, Chang Zhang, Xi Zhu, Chenguang Huang, Yu Tong, Bing Xu, Jianfei Bai, Jian Tong, et al. Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in Neural Information Processing Systems*, 33:17766–17778, 2020.

Hao Chen, Gonzalo E. Constante Flores, and Can Li. Physics-informed neural networks with hard linear equality constraints. *Computers Chemical Engineering*, 189:108764, 2024. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2024.108764.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International conference on machine learning*, pages 883–892. PMLR, 2018.

Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. Problog: A probabilistic prolog and its application in link discovery. In *IJCAI 2007, Proceedings of the 20th international joint conference on artificial intelligence*, pages 2462–2467. IJCAI-INT JOINT CONF ARTIF INTELL, 2007.

Luca Di Liello, Pierfrancesco Ardino, Jacopo Gobbi, Paolo Morettin, Stefano Teso, and Andrea Passerini. Efficient generation of structured objects with constrained adversarial networks. *Advances in neural information processing systems*, 33:14663–14674, 2020.

Persi Diaconis and Sandy Zabell. Closed form summation for classical distributions: variations on a theme of de moivre. *Statistical Science*, pages 284–302, 1991.

Michelangelo Diligenti, Marco Gori, Marco Maggini, and Leonardo Rigutini. Bridging logic and kernel machines. *Machine learning*, 86:57–88, 2012.

Josip Djolonga and Andreas Krause. Differentiable learning of submodular models. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/192fc044e74dffea144f9ac5dc9f3395-Paper.pdf.

Priya Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning in stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 5484–5494, 2017.

Marc Fischer, Mislav Balunovic, Dana Drachsler-Cohen, Timon Gehr, Ce Zhang, and Martin Vechev. Dl2: training and querying neural networks with logic. In *International Conference on Machine Learning*, pages 1931–1941. PMLR, 2019.

Gonzalo E Constante Flores, Hao Chen, and Can Li. Enforcing hard linear constraints in deep learning models with decision rules. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Artur d'Avila Garcez and Luis C Lamb. Neurosymbolic ai: The 3 rd wave. *Artificial Intelligence Review*, 56(11):12387–12406, 2023.

Eleonora Giunchiglia and Thomas Lukasiewicz. Multi-label classification neural networks with hard logical constraints. *Journal of Artificial Intelligence Research*, 72:759–818, 2021.

Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. Stochastic optimization of sorting networks via continuous relaxations. In *International Conference on Learning Representations*, 2019.

Jiawei He, Yu Gong, Joseph Marino, Greg Mori, and Andreas Lehrmann. Variational autoencoders with jointly optimized latent dependency structure. In *International conference on learning representations*, 2018.

Johannes N. Hendriks, Carl Jidling, Adrian G. Wills, and Thomas Bo Schön. Linearly constrained neural networks. *ArXiv*, abs/2002.01600, 2020.

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

Andrew Ilyas, Logan Engstrom, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. A closer look at deep policy gradients, 2020. URL https://arxiv.org/abs/1811.02553.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Kijung Jeon, Michael Muehlebach, and Molei Tao. Fast non-log-concave sampling under nonconvex equality and inequality constraints with landing. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Carolyn Kim, Ashish Sabharwal, and Stefano Ermon. Exact sampling with integer linear programs and random perturbations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, 2009.

Matieyendou Lamboni. Efficient dependency models: Simulating dependent random variables. *Mathematics and Computers in Simulation*, 200:199–217, 2022. ISSN 0378-4754. doi: https://doi.org/10.1016/j.matcom.2022.04.018.

Huichen Li, Linyi Li, Xiaojun Xu, Xiaolu Zhang, Shuang Yang, and Bo Li. Nonlinear projection based gradient estimation for query efficient blackbox attacks. In *International conference on artificial intelligence and statistics*, pages 3142–3150. PMLR, 2021.

Anji Liu, Mathias Niepert, and Guy Van den Broeck. Image inpainting via tractable steering of diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738, 2015.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Eleonora Misino, Giuseppe Marra, and Emanuele Sansone. Vael: Bridging variational autoencoders and probabilistic logic programming. *Advances in Neural Information Processing Systems*, 35:4667–4679, 2022.

Sai Shankar Narasimhan, Shubhankar Agarwal, Litu Rout, Sanjay Shakkottai, and Sandeep P Chinchali. Constrained posterior sampling: Time series generation with hard

constraints. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Ali Raza, Arni Sturluson, Cory M Simon, and Xiaoli Fern. Message passing neural networks for partial charge assignment to metal–organic frameworks. *The Journal of Physical Chemistry C*, 124(35):19070–19082, 2020.

William F. Sharpe. Mutual fund performance. *Journal of Business*, 39(1):119–138, 1966.

Alexander Shekhovtsov, Viktor Yanush, and Boris Flach. Path sample-analytic gradient estimators for stochastic binary networks. *Advances in neural information processing systems*, 33:12884–12894, 2020.

Vinay Shukla, Zhe Zeng, Kareem Ahmed, and Guy Van den Broeck. A unified approach to count-based weakly supervised learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Neural Information Processing Systems*, 2016. URL https://api.semanticscholar.org/CorpusID:10447416.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pages 11895–11907, 2019.

Mihaela C Stoian, Salijona Dyrmishi, Maxime Cordy, Thomas Lukasiewicz, and Eleonora Giunchiglia. How realistic is your synthetic data? constraining deep generative models for tabular data. In *The Twelfth International Conference on Learning Representations*, 2024.

Mihaela Catalina Stoian, Eleonora Giunchiglia, and Thomas Lukasiewicz. Exploiting t-norms for deep learning in autonomous driving. In Artur S. d'Avila Garcez, Tarek R. Besold, Marco Gori, and Ernesto Jiménez-Ruiz, editors, *Proceedings of the 17th International Workshop on Neural-Symbolic Learning and Reasoning, NeSy 2023, La Certosa di Pontignano, Siena, Italy, 3–5 July 2023*, pages 369–380, July 2023.

Sebastian Tschiatschek, Aytunc Sahin, and Andreas Krause. Differentiable submodular maximization. In *International Joint Conference on Artificial Intelligence*, 2018.

Frédéric Vrins. Sampling the multivariate standard normal distribution under a weighted sum constraint. *Risks*, 6(3), 2018. ISSN 2227-9091. doi: 10.3390/risks6030064.

Runzhong Wang, Yunhao Zhang, Ziao Guo, Tianyi Chen, Xiaokang Yang, and Junchi Yan. LinSATNet: The positive linear satisfiability neural networks. In *International Conference on Machine Learning (ICML)*, 2023.

Bryan Wilder. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 2019.

Sang Michael Xie and Stefano Ermon. Reparameterizable subset sampling via continuous relaxations. *International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Broeck. A semantic loss function for deep learning with symbolic knowledge. In *International conference on machine learning*, pages 5502–5511. PMLR, 2018.

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1–10, 2015.

Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Zhe Zeng, Paolo Morettin, Fanqi Yan, Antonio Vergari, and Guy Van den Broeck. Probabilistic inference with algebraic constraints: Theoretical limits and practical approximations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 11564–11575. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/85934679f30131d812a8c7475a7d0f74-Paper.pdf.

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang, and Guy Van den Broeck. On the paradox of learning to reason from data. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 3365–3373, 2023.

Ziyao Zhang, Stefan Zohren, and Stephen Roberts. Deep learning for portfolio optimization. *The Journal of Financial Data Science*, 2(4):8–20, 2020.

Appendix A. Limitations

Gaussian assumption is widely adopted in generative modeling, such as in VAEs and diffusion models and is shared among scientific applications. Such assumptions do not undermine the practical relevance of our framework.

We do agree that it is an interesting future direction to extend our work to other distributions. To highlight the potential, in Appendix I, we generalized our theoretical results to Poisson distributions, the most commonly used distributional assumption for categorical variables, and demonstrated that in the Poisson setting, our proposed gradient estimator remains effective, showing the broad applicability and robustness of our work.

Another interesting future direction is to consider a broader class of constraints, for example, the algebraic constraints. However, Zeng et al. (2020) shows that the constrained distribution under such constraints becomes intractable, and thus, an extension to such constraints is non-trivial, which we look forward to addressing in our future work.

Appendix B. Broader Impacts

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

Appendix C. Additional Experiment Details in Synthetic Settings

We carried out a series of experiments to analyze the effectiveness of our gradient estimator from Gaussian variable. Our focus lies on three pivotal metrics: bias, variance, and the average error. Since, we only care about the direction of the gradients, we employed the cosine distance, namely 1 – cosine similarity, to measure the deviation of our gradient estimators from the ground truth vector. The ground truth are sampled from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ satisfying the constraint. We randomly generated 20 sets of parameters and calculated the metrics for each set. Then, we take average of these 20 repeats and computed their standard deviations. The randomly generated gradients are sampled from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

- **bias**: $1 - cos\left(\frac{\sum_j^n h_j}{n}, h_{gt}\right)$
- **variance**: $var\left(\left\{1 - cos\left(h_i, \frac{\sum_j^n h_j}{n}\right)\right\}_{i=1}^{n}\right)$
- **averaged error**: $\frac{\sum_{i=1}^n 1 - cos(h_i, h_{gt})}{n}$

where h_i denotes the approximated gradient and h_{gt} denotes the ground truth gradient.

Appendix D. Additional Experimental Details for VAE Constrained Latent Space

Model We present the model architecture used in the experiment.
Encoders:

$$\text{fc}(\text{input\_size}, 512) \to \text{ReLU} \to \text{fc}(512, 256) \to \text{ReLU} \to \text{fc}(512, z_{dim})$$

Sampling: We use two separate $\text{fc}(z_{dim}, z_{dim})$ for predicting the mean and log variance of the latent distribution. We sample exactly from the constrained distribution.
Decoders:

$$\text{fc}(z_{dim}, 256) \to \text{ReLU} \to \text{fc}(256, 512) \to \text{ReLU} \to \text{fc}(512, \text{input\_size}) \to \texttt{output\_function}$$

`output_function` is `sigmoid()` predicting the mean of Bernoulli Observations.

Training All models were implemented with PyTorch and trained using the Adam optimizer with a mini-batch size of 128 and learning rate 0.0001. All models are trained with 100 epochs.

D.1. Experiment Results

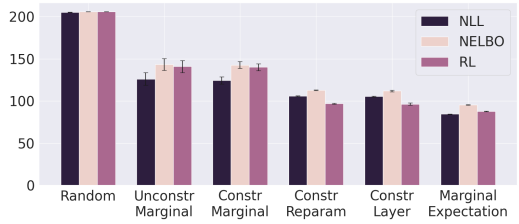

Figure 3: Comparison of gradient estimators for VAE with constrained latent space. Negative log-likelihood (NLL), negative ELBO (NELBO), and reconstruction loss (RL) are averaged over 5 trials.

Appendix E. Additional Experimental Details for VAE Constrained Data Generation

Table 7: Comparison on VAE Constraint Violation and Training Time. The tabel reports the average training time for one epoch. We observe that our approach to enforce the constraints does not cause significant increase in training time. For Vanilla VAE and Ladder VAE, the increase in training time is less than 1 seconds.

Algorithm	Training Time ↓
VAE	$\mathbf{3.94 \pm 0.14}$
Constrained VAE	4.21 ± 0.11
Ladder VAE	$\mathbf{10.11 \pm 0.37}$
Constrained Ladder VAE	10.90 ± 0.47
Graph VAE	$\mathbf{44.11 \pm 0.50}$
Constrained Graph VAE	46.61 ± 0.59

Model We adopted a model architecture similar to (He et al., 2018).

Encoders:

$$\text{fc}(\text{input\_size}, 512) \rightarrow \text{batch\_norm} \rightarrow \text{ELU} \rightarrow \text{fc}(512, 512) \rightarrow \text{batch\_norm} \rightarrow \text{ELU} \rightarrow \text{fc}(512, 256) \rightarrow \text{batch\_norm} \rightarrow \text{ELU} \rightarrow \text{fc}(256, 128)$$

Decoders:

$$\text{fc}(N', 256) \rightarrow \text{batch\_norm} \rightarrow \text{ELU} \rightarrow \text{fc}(256, 512) \rightarrow \text{batch\_norm} \rightarrow \text{ELU} \rightarrow \text{fc}(512, 512) \rightarrow \text{batch\_norm} \rightarrow \text{ELU} \rightarrow \text{fc}(512, \text{input\_size}) \rightarrow \text{output\_function}()$$

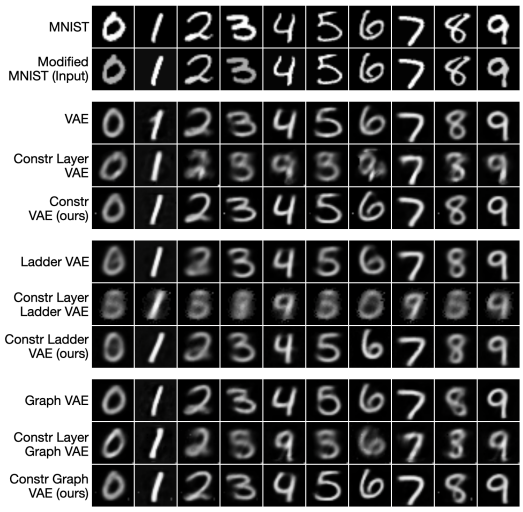

Figure 4: The first block displays the original MNIST images and the ones modified by the brightness constraint as inputs. For the following blocks, each displays the reconstructed images by different VAE architectures. Within each block, the first row is generated by the unconstrained VAE, the second by VAE constrained by the baseline Constrained Layer and the last one by VAE constrained by our method.

`output_function` is `sigmoid()` predicting μ, `fc`(`input_size`, `input_size`) predicting log var of Gaussian observations.

Training All models were implemented with PyTorch and trained using the Adam optimizer with a mini-batch size of 128. We conducted hyperparameter tuning on learning rate and adopt the best learning rate in the final model. All models are trained with 1000 epochs.

Dataset We modify the original MNIST dataset by introducing linear equality constraint on every image. We constraint the sum of all pixel values in every image to be 100, which is approximately the mean and median of all images in MNIST. We first scale the pixel values so the sum is 100. To enforce the pixel values in the range $[0, 1]$, we uniformly distribute the extra pixel values to white pixels.

Training Time In order to show that *Marginal Expectation* adds minimal training time, we measure the training time of 1 epoch and report the average training time. We record the average training time of all the models for one epoch. All results are averaged over 5 independent runs. The results are summarized in Table 2. The results show that the constrained versions are less than 10% slower than the unconstrained version. *Marginal Expectation* adds less than 1 second of additional training time for VAE and Ladder VAE.

Appendix F. Additional Experimental Detail for Diffusion Model Constrained Data Generation

F.1. Algorithm

We present the algorithm for incorporating our exact sampling method in selected intermediate backward diffusion steps in Algorithm 1.

Algorithm 1 DDIM with Exact Sampling during Inference.

Require: $\tau = \{\tau_0, \tau_1, ..., \tau_K\}$ (diffusion timestamp sequence, where $\tau_0 = 0, \cdots \tau_K = T$.

$\boldsymbol{x}_{\tau_K} \sim \mathcal{N}(0, I)$
for $i = K, K-1, \ldots, 1, 0$ **do**
  if Exact Sampling **then**
    $\boldsymbol{x}_0^{\tau_{i-1}} \sim \mathcal{N}(\boldsymbol{x}_0^{\tau_{i-1}}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_{\tau_i}, t), \boldsymbol{\Sigma}_\theta, \boldsymbol{A}\boldsymbol{\mu}_\theta = \boldsymbol{k})$
  else
    $\boldsymbol{x}_0^{\tau_{i-1}} \sim \mathcal{N}(\boldsymbol{x}_0^{\tau_{i-1}}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_{\tau_i}, t), \boldsymbol{\Sigma}_\theta)$
  end if
  $\boldsymbol{x}_{\tau_{i-1}} \sim p(\boldsymbol{x}_{\tau_{i-1}} \mid \boldsymbol{x}_0^{\tau_{i-1}}, \boldsymbol{x}_{\tau_i})$
end for
Return $\boldsymbol{x}_{\tau_0}$

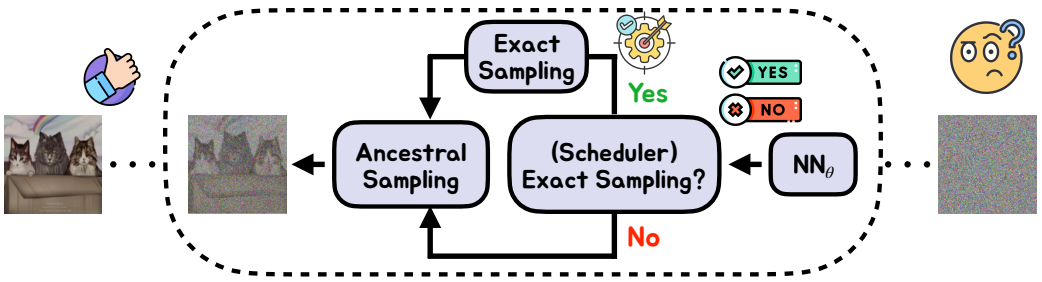

Figure 5: The inference pipeline of diffusion models involves exact sampling at the final step of backward diffusion for DDPM, whereas for DDIM, exact sampling is conducted at selected backward diffusion steps, as investigated in the following sections.

F.2. Model Architecture

The denoising model used in DDPM is a U-Net (Ho et al., 2020) architecture. The details of the model used for CIFAR 10 are as follows:

- **Channels** (*ch*): 128
- **Channel Multipliers** (*ch_mult*): {1, 2, 2, 2}
- **Dropout**: 0.1
- **Number of Residual Blocks** (*num_res_blocks*): 2

- **Number of Attention Blocks** (*attn*): 2

The details of the model used for CELEBA, LSUN Church, and LSUN Cat are as follows:

- **Channels** (*ch*): 128
- **Channel Multipliers** (*ch_mult*): {1, 1, 2, 2, 4}
- **Dropout**: 0.1
- **Number of Residual Blocks** (*num_res_blocks*): 2
- **Number of Attention Blocks** (*attn*): 2

F.3. Training and Evaluation

We trained the DDPM using a distributed setup on NVIDIA V100 GPUs with 32GB of memory. For the CIFAR-10 dataset, we used 4 GPUs, while training on CELEBA-HQ, LSUN Church, and LSUN Cat utilized 6 GPUs. Training was conducted with exponential moving average (EMA) applied to the model parameters, using a decay factor of 0.9999. The number of diffusion steps was fixed at $T = 1000$ without a hyperparameter sweep, employing a linear schedule for the noise variance from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. CIFAR-10 was trained for 800,000 steps, CelebA-HQ for 500,000 steps, LSUN Cat for 1.8 million steps, and LSUN Church for 1.2 million steps. Images from CelebA-HQ, LSUN Cat, and LSUN Church datasets were uniformly downsampled to 128×128 resolution.

For evaluation, Inception and Fréchet Inception Distance (FID) scores were calculated on 50,000 samples.

F.4. Data

We modify the original datasets by introducing linear equality constraint on every channel for each image. We constraint the sum of all pixel values in every image to be the mean or median of all images in the dataset.

F.5. DDIM Experiment Results

F.6. Additional Experiment Results

We first explore what would be the optimal schedule policy by comparing against: 1.) *Uniform N*: Distributing N constrained exact sampling evenly across diffusion steps; 2.) *Start M, End N* Placing M consecutive constrained exact sampling at the start and N at the end of the diffusion process; 3.) *Start N, Space S*: Placing N constrained exact sampling at the start with a spacing of S; 4.) *End N, Space S*: Placing N constrained exact sampling at the end with a spacing of S. The results are shown in Table 9, which suggest that placement at the end exactly enforces constraint satisfaction, while constraint layers placed at the start produce higher IS. For optimal balance, we adopt the *Start N End N* schedule, which places N correction steps at both the beginning and the end of the diffusion process. Next, we test the optimal number of N under this schedule in Table 10.

Since $N = 2$ achieves the lowest FID and lowest IS and $N = 4$ achieves highest FID and highest IS, we adopt $N = 3$ to balance these two metrics.

Table 8: Comparison of constrained and unconstrained models across datasets using DDIM sampling.

Dataset	Model	FID ↓	IS ↑	Violation ↓
CIFAR	Ours	**7.972**	8.585 ± 0.118	**0**
	DDIM	8.123	**8.646 ± 0.084**	1.0
CelebA	Ours	**12.389**	**2.367 ± 0.023**	**0**
	DDIM	12.417	2.366 ± 0.033	0.999
LSUN Church	Ours	**6.557**	**2.596 ± 0.033**	**0**
	DDIM	6.702	2.584 ± 0.027	0.9999
LSUN Cat	Ours	**18.902**	**4.857 ± 0.037**	**0**
	DDIM	18.958	4.849 ± 0.062	0.9999

Table 9: Comparison of schedules based on FID, IS, and Violation metrics. The experiments are conducted on CIFAR-10 dataset.

Schedule	FID ↓	IS ↑	Violation ↓
Uniform 4	7.979	8.589 ± 0.121	0
Start 3 End 1	8.049	8.612 ± 0.136	0
Start 2 End 2	7.823	8.570 ± 0.069	0
Start 1 End 3	7.912	8.616 ± 0.088	0
Start 4 Space 1	7.999	8.670 ± 0.108	0.9999
Start 4 Space 2	8.040	8.580 ± 0.102	0.9999
Start 4 Space 3	8.092	8.564 ± 0.080	0.9999
End 4 Space 1	8.061	8.580 ± 0.098	0
End 4 Space 2	7.924	8.603 ± 0.096	0
End 4 Space 3	8.016	8.594 ± 0.133	0

Appendix G. Additional Experimental Details for Partial Charge Predictions

Training Here, we describe our training and evaluation process for the exact-k constrained MPNN. We conducted a random partitioning of the dataset containing 2266 charge-labeled MOFs, creating distinct training, validation, and test sets (70/10/20%). We use the training set for direct model parameter tuning, while the validation set determines stopping criteria. The test set plays a crucial role in providing an unbiased assessment of the final model's performance.

Hyperparameter Tuning To optimize our model's performance, we conduct a systematic hyperparameter tuning process, sequentially optimizing six key hyperparameters: Learning rate, Batch size, Time steps, Embedding size, Hidden Feature size, and Patience Threshold. The optimal hyperparameter values are reported in supplementary materials.

The optimal hyperparameter values for closed-form expected loss are: lr = 0.005, batch size = 128, time steps = 6, embedding size = 20, hidden feature size = 50, and patience threshold = 300, and the optimal hyperparameter values for negative log likelihood loss are

Table 10: Performance metrics across different N under the Start N End N schedule. The experiments are conducted on CIFAR-10 dataset.

N	FID ↓	IS ↑	Violation ↓
1	7.977	8.608 ± 0.086	0
2	7.823	8.570 ± 0.069	0
3	7.972	8.585 ± 0.118	0
4	8.007	8.671 ± 0.087	0
5	8.004	8.591 ± 0.104	0
6	7.975	8.578 ± 0.081	0

: lr = 0.005, batch size = 128, time steps = 5, embedding size = 30, hidden feature size = 50, and patience threshold = 300.

Appendix H. Additional Experimental Details for Chemical Process Units and Subsystems

Experiment setting In a Continuous Stirred-Tank Reactor (**CSTR**) benzene and ethylene react to produce ethylbenzene following $B + E \rightarrow EB$. The stoichiometric relationship ensures reactant consumption matches product formation, governed by linear equality constraints. We consider the setting where the reactor operates with fixed volume and pressure, leaving the molar flow rates of benzene, ethylene, and the working temperature as design variables. A neural surrogate model is trained to predict output flow rates. In a chemical plant (**plant**) that produces DME and DEE uses methanol, ethanol, and water as feed, Mass balance ensures consistency between inflows, reactions, outflows, and recycling streams. Assuming the entire system is fixed, a surrogate model can predict output flow based on inputs and recycling. In an extractive distillation subsystem (**distillation**) which separates a 50/50 azeotropic mixture of n-heptane and toluene using phenol as a solvent, two distillation columns separate n-heptane at the top and toluene with phenol solvent recovered at the bottom. A surrogate model is developed to predict heat duties and the flow rates of components in the distillate streams for optimal operating conditions. With no chemical reactions, the sum of the molar flow rates of n-heptane, toluene, and phenol always equals the distillate rate.

We specify the input and output variables as well as governing constraints for the three experiments. x_i denotes the input variables and y_i denotes the output variables.

CSTR

- x_1: Temperature of the RX unit.
- x_2: Molar flow rate of Benzene in the B stream.
- x_3: Molar flow rate of Ethylene in the E stream.
- y_1: Molar flow rate of Ethylbenzene in the EB stream.
- y_2: Molar flow rate of Benzene in the EB stream.

- y_3: Molar flow rate of Ethylene in the EB stream.

The linear equality constraints governing the system are:

$$-y_2 + y_3 = -x_2 + x_3, \quad \text{Reactants consumption},$$

$$-y_1 - y_2 = -x_2, \quad \text{EB production}.$$

plant

- x_1: Mass flow rate of methanol in the FEED stream.
- x_2: Mass flow rate of ethanol in the FEED stream.
- x_3: Mass flow rate of water in the FEED stream.
- x_4: Total mass flow rate of the PURGE stream.
- y_1: Total mass flow rate of the DME stream.
- y_2: Mass flow rate of DME in the DME stream.
- y_3: Total mass flow rate of the DEE stream.
- y_4: Mass flow rate of DEE in the DEE stream.
- y_5: Total mass flow rate of the WATER stream.

The system satisfies the following mass balance equation:

$$-y_1 - y_3 - y_5 = -x_1 - x_2 - x_3 + x_4, \quad \text{Mass balance}.$$

distillation

- x_1: Molar flow rate of phenol in the SOLVENT stream.
- x_2: Reflux ratio of COLUMN column.
- x_3: Distillate rate of COLUMN column.
- x_4: Reflux ratio of COL-REC column.
- x_5: Distillate rate of COL-REC column.
- y_1: Molar flow rate of n-heptane in the C7 stream.
- y_2: Molar flow rate of toluene in the TOLUENE stream.
- y_3: Condenser heat duty of COLUMN column.
- y_4: Reboiler heat duty of COLUMN column.
- y_5: Condenser heat duty of COL-REC column.
- y_6: Reboiler heat duty of COL-REC column.

- y_7: Molar flow rate of toluene in the C7 stream.
- y_8: Molar flow rate of phenol in the C7 stream.
- y_9: Molar flow rate of n-heptane in the TOLUENE stream.
- y_{10}: Molar flow rate of phenol in the TOLUENE stream.

The system satisfies the following linear equality constraints:

$$-y_1 - y_7 - y_8 = -x_3, \quad \text{C7 fractions},$$

$$-y_2 - y_9 - y_{10} = -x_5, \quad \text{TOLUENE fractions}.$$

Training and Validation Loss Curve We present the training and validation loss curve, which demonstrates the faster convergence of our approach.

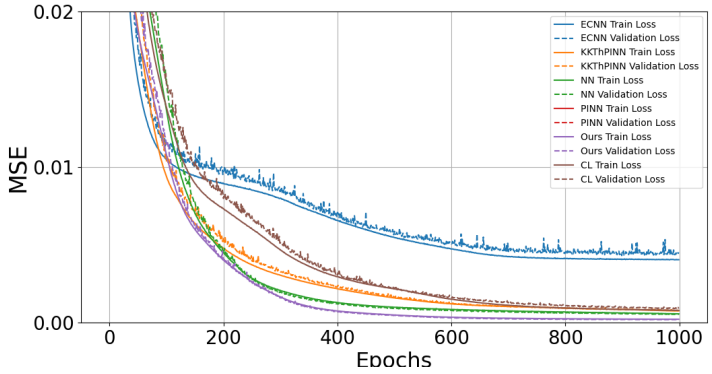

Figure 6: Training and validation MSE loss curve for **CSTR**. All results are averaged over 10 independent runs.

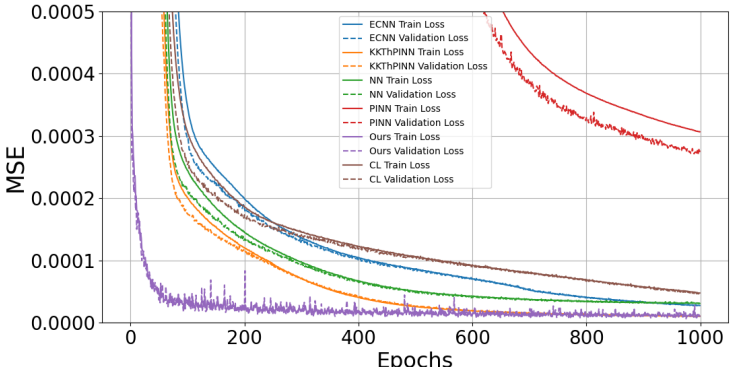

Figure 7: Training and validation MSE loss curve for **plant**. All results are averaged over 10 independent runs.

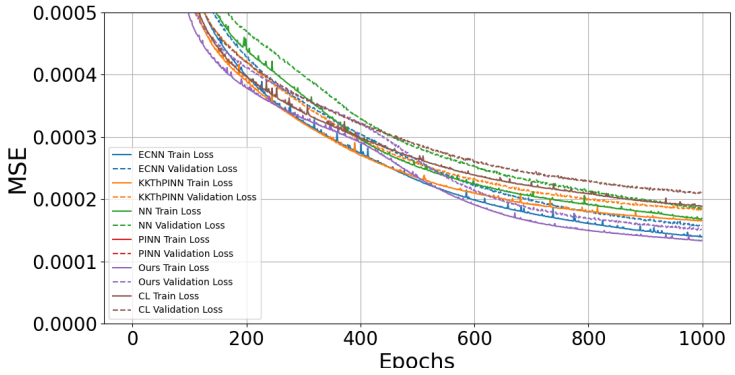

Figure 8: Training and validation MSE loss curve for **distillation**. All results are averaged over 10 independent runs.

Appendix I. Beyond Gaussian

In this section, we present the theoretical results when $\boldsymbol{z}$ are Poisson variables defined over discrete domains. Similar to the Gaussian setting, we find that when the element-wise loss ℓ is $L1$ or $L2$ loss and the mapping $f_{\boldsymbol{u}}$ is an identity function, the expected loss admits closed-form expressions.

Proposition 3 (Poisson Closed-form Expected Loss) *Let $\boldsymbol{z} = (z_1, \ldots, z_n)^T$, where $z_i \sim Poisson(\theta_i)$. Let $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ be the ground truth vector subject to the equality constraint $\sum_{i=1}^n y_i = k$ with $k \in \mathbb{N}^+$. Then it holds that*

i) when ℓ is L1 loss,

$$\begin{aligned} L(\boldsymbol{\theta}) = \sum_{i=1}^{n} & (k - \lfloor kp_i - d_i \rfloor) p_i Bin(\lfloor kp_i - d_i \rfloor; k, p_i) \\ & + \lfloor kp_i - d_i \rfloor (1 - p_i) Bin(\lfloor kp_i - d_i \rfloor; k, p_i) \\ & - 2d_i F(\lfloor kp_i - d_i \rfloor; k, p_i) + d_i; \end{aligned}$$

ii) when ℓ is L2 loss, $L(\boldsymbol{\theta}) = \sum_{i=1}^n kp_i(1 - p_i) + k^2 p_i^2 - 2y_i kp_i + y_i^2$,
where Bin denotes the probability mass function (p.m.f.) of a binomial distribution and F denotes a regularized incomplete beta function. $d_i = kp_i - y_i$ and $p_i = \frac{\theta_i}{\sum_{j=1}^n \theta_j}$.

In the general setting when there is no closed-form solution for the expected loss, we first show that exact sampling from the constrained distribution can be achieved as it takes its form as a multinomial distribution.

Proposition 4 (Poisson Constrained Distribution) *Given $\boldsymbol{z} = (z_1, \ldots, z_n)^T$ with $z_i \sim Poisson(\theta_i)$, the constrained distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \sum_{j=1}^n z_n = k)$ is equivalent to a multinomial distribution with parameter k and probabilities $\frac{\theta_1}{\sum_{j=1}^n \theta_j}, \ldots, \frac{\theta_n}{\sum_{j=1}^n \theta_j}$.*

In the backward pass, the results below allow us to derive gradient estimators with either the conditional marginals or the expectation of the conditional marginal as a differentiable proxy.

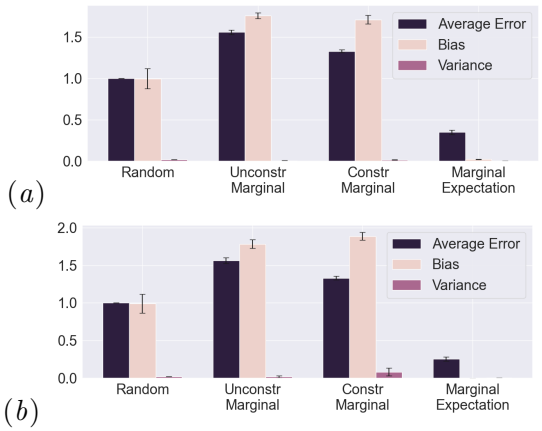

Figure 9: Comparisons of different gradient estimators for point-wise loss ℓ being L1 and L2 loss applied to Poisson variable are conducted. The comparison follows the same setting as Gaussian variables. We show that our proposed gradient estimator *Marginal Expectation* outperforms baselines by a significant margin.

Proposition 5 (Poisson Conditional Marginal and Expectations) *Given* $\boldsymbol{z} = (z_1, \ldots, z_n)^T$ *with* $z_i \sim Poisson(\theta_i)$*, the conditional marginal* $p_{\boldsymbol{\theta}}(z_i \mid \sum_{j=1}^n z_j = k)$ *follows a binomial distribution with parameter* k *and probability* $\frac{\theta_i}{\sum_{j=1}^n \theta_j}$*. Further, its expectation is* $\frac{k\theta_i}{\sum_{j=1}^n \theta_j}$*.*

We conduct Synthetic Experiment for Poisson variables with settings similar to those of Gaussian variables. We compare our gradient estimator *Marginal Expectation* with three baselines, namely *Random*, *Unconstrained Marginal*, as well as *Constrained Marginal*. Results are shown in 9. We observe similar results as in the Gaussian settings. Estimator *Unconstrained Marginal* and *Random* have similar performances. *Constrained Marginal* still has similarly bad performances as *Random*, while *Marginal Expectation* outperforms these baselines by a noticeable margin. We show that our *Marginal Expectation* is not limited to Gaussian and Bernoulli variables.

Appendix J. Proofs

J.1. Proposition 6

Proposition 6 (Gaussian Constrained Distribution) *Given* $\boldsymbol{z} = (z_1, \ldots, z_n)^T \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$*, the constrained distribution* $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{A}\boldsymbol{z} = \boldsymbol{k})$ *is equivalent to an* $n - a$ *dimensional multivariate Gaussian distribution with mean* $\overline{\boldsymbol{\mu}} \in \mathbb{R}^{n-a}$ *and covariance matrix* $\overline{\boldsymbol{\Sigma}} \in \mathbb{R}^{n-a \times n-a}$ *defined as*

$$\overline{\boldsymbol{\mu}} = \boldsymbol{E}\boldsymbol{\mu} + \boldsymbol{E}\boldsymbol{\Sigma}\boldsymbol{A}^T \left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1} (\boldsymbol{k} - \boldsymbol{A}\boldsymbol{\mu}),$$
$$\overline{\boldsymbol{\Sigma}} = \boldsymbol{E}\boldsymbol{\Sigma}\boldsymbol{E}^T - \boldsymbol{E}\boldsymbol{\Sigma}\boldsymbol{A}^T \left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1} \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{E}^T.$$

where $\boldsymbol{E} \in \mathbb{R}^{(n-a)\times(n)}$ *is the first* $n - a$ *rows of an identity matrix* $\boldsymbol{I} \in \mathbb{R}^{n \times n}$*.*

Proof Let $\boldsymbol{z} = (z_1, \ldots, z_n)^T$ with $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Define $\boldsymbol{A}$ be the constraint matrix with $\text{rank}(\boldsymbol{A}) = a < n$. Define $\boldsymbol{E} \in \mathbb{R}^{(n-a)\times(n)}$ to contain the first $n-a$ rows of an identity matrix $\boldsymbol{I} \in \mathbb{R}^{n\times n}$. Thus, we consider the following linear transformation defined by $\boldsymbol{C}$.

$$\boldsymbol{C}\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{E}\boldsymbol{\mu} \\ \boldsymbol{A}\boldsymbol{\mu} \end{pmatrix}$$
$$\boldsymbol{C}\boldsymbol{\Sigma}\boldsymbol{C}^T = \begin{pmatrix} \boldsymbol{E}\boldsymbol{\Sigma}\boldsymbol{E}^T & \boldsymbol{E}\boldsymbol{\Sigma}\boldsymbol{A}^T \\ \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{E}^T & \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T \end{pmatrix}$$

Thus, the conditional distribution of $\boldsymbol{E}\boldsymbol{z}$ condition on $\boldsymbol{A}\boldsymbol{z}$ follows multivariate Gaussian distribution with parameters as follows:

$$\overline{\boldsymbol{\mu}} = \boldsymbol{E}\boldsymbol{\mu} + \boldsymbol{E}\boldsymbol{\Sigma}\boldsymbol{A}^T \left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1} (\boldsymbol{k} - \boldsymbol{A}\boldsymbol{\mu})$$
$$\overline{\boldsymbol{\Sigma}} = \boldsymbol{E}\boldsymbol{\Sigma}\boldsymbol{E}^T - \boldsymbol{E}\boldsymbol{\Sigma}\boldsymbol{A}^T \left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1} \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{E}^T$$

■

J.2. Proposition 7

Proposition 7 (Gaussian Closed-form Expected Loss) *Let $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Let $\boldsymbol{y} = (y_1, \ldots, y_n)^T$ be the ground truth vector subject to the equality constraint $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{k}$. Then it holds that*

i) when ℓ is L1 loss, $L(\boldsymbol{\theta})$ has closed form
$\sum_{i=1}^n \overline{\boldsymbol{\Sigma}}_{i,i} \sqrt{\frac{2}{\pi}} e^{-\frac{(\overline{\boldsymbol{\mu}}_i - y_i)^2}{2\overline{\boldsymbol{\Sigma}}_{i,i}^2}} + (\overline{\boldsymbol{\mu}}_i - y_i)\, erf\left(\frac{\overline{\boldsymbol{\mu}}_i - y_i}{\sqrt{2}\,\overline{\boldsymbol{\Sigma}}_{i,i}}\right)$;

ii) when ℓ is L2 loss, $\sum_{i=1}^n \overline{\boldsymbol{\mu}}_i^2 + \overline{\boldsymbol{\Sigma}}_{i,i}^2 - 2y_i\overline{\boldsymbol{\mu}}_i + y_i^2$,

where $\overline{\boldsymbol{\mu}}$ and $\overline{\boldsymbol{\Sigma}}$ are defined above.

Proof Let $\boldsymbol{z} = (z_1, \ldots, z_n)^T \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Let $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ be the ground truth subject to the equality constraint $\boldsymbol{A}\boldsymbol{y} = \boldsymbol{k}$. We derive a closed-form solution for the L1 loss of $\boldsymbol{z}$ subject to the constraint $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{k}$.

$$\begin{aligned} L(\theta) &= \mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z}|\boldsymbol{A}\boldsymbol{z}=\boldsymbol{k})}[\| \boldsymbol{z} - \boldsymbol{y} \|] \\ &= \sum_{i=1}^n \mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z}|\boldsymbol{A}\boldsymbol{z}=\boldsymbol{k})}[\| z_i - y_i \|] \end{aligned}$$

We know that $z_i \sim N\left(\overline{\mu}_i, \overline{\sigma}_i^2\right)$ from below. Then, define $l_i = z_i - y_i$. Then $l_i \sim N\left(\overline{\mu}_i - y_i, \overline{\sigma}_i^2\right)$. Thus, $\mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z}|\boldsymbol{A}\boldsymbol{z}=\boldsymbol{k})}[| l_i |]$ is the mean of a folded normal distribution.

$$\begin{aligned} \mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z}|\boldsymbol{A}\boldsymbol{z}=\boldsymbol{k})}[| l_i |] = \overline{\sigma}_i^2 \sqrt{\frac{2}{\pi}} \exp\left(\frac{-(\overline{\mu}_i - y_i)^2}{2\overline{\sigma}_i^2}\right) \\ +(\overline{\mu}_i - y_i) erf\left(\frac{\overline{\mu}_i - y_i}{\sqrt{2\overline{\sigma}_i^2}}\right) \end{aligned}$$

We also derive a closed-form solution for the L2 loss of $\boldsymbol{z}$ subject to the constraint $\boldsymbol{Az} = \boldsymbol{k}$.

$$\begin{aligned}
L(\theta) &= \mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z}|\boldsymbol{Az}=\boldsymbol{k})}[\| \boldsymbol{z} - \boldsymbol{y} \|^2] \\
&= \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{z}\sim p_\theta(z_i|\boldsymbol{Az}=\boldsymbol{k})}[\| z_i - y_i \|^2] \\
&= \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{z}\sim p_\theta(z_i|\boldsymbol{Az}=\boldsymbol{k})}[z_i^2 - 2y_i z_i + y_i^2] \\
&= \sum_{i=1}^{n} \overline{\mu}_i + \overline{\sigma}_i^4 - 2y_i\overline{\mu}_i + y_i^2
\end{aligned}$$

■

J.3. Proposition 8

Proposition 8 (Gaussian Conditional Marginal and Expectations) *Given* $\boldsymbol{z} = (z_1, \ldots, z_n)^T \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$*, the conditional marginal* $p_{\boldsymbol{\theta}}(z_i \mid \boldsymbol{Az} = \boldsymbol{k})$ *follows a univariate Gaussian distribution with mean* $\overline{\mu}_i = \mu_i + \boldsymbol{e}_i^T\boldsymbol{\Sigma}\boldsymbol{A}\left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1}(\boldsymbol{k} - \boldsymbol{A}\boldsymbol{\mu})$ *and variance* $\overline{\sigma}_i^2 = \boldsymbol{e}_i^T\boldsymbol{\Sigma}\boldsymbol{e}_i - \boldsymbol{e}_i^T\boldsymbol{\Sigma}\boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1}\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{e}_i$*. Further, the expectation of the marginal distribution is* $\overline{\mu}_i$*.*

Proof To derive the marginal distribution, let's consider the standard basis vector $\boldsymbol{e}_i$ in $\mathbb{R}^n$. Define the linear transformation $\boldsymbol{B}$ such that

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{e}_i^T \\ \boldsymbol{A} \end{pmatrix}$$

Then,

$$\begin{aligned}
\boldsymbol{B}\boldsymbol{\mu} &= \begin{pmatrix} \mu_i \\ \boldsymbol{A}\boldsymbol{\mu} \end{pmatrix} \\
\boldsymbol{B}\boldsymbol{\Sigma}\boldsymbol{B}^T &= \begin{pmatrix} \boldsymbol{e}_i^T\boldsymbol{\Sigma}\boldsymbol{e}_i & \boldsymbol{e}_i^T\boldsymbol{\Sigma}\boldsymbol{A}^T \\ \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{e}_i & \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T \end{pmatrix}
\end{aligned}$$

Thus, the conditional distribution of z_i condition on $\boldsymbol{Az}$ follows multivariate normal distribution with parameters as follows:

$$\begin{aligned}
\overline{\mu}_i &= \mu_i + \boldsymbol{e}_i^T\boldsymbol{\Sigma}\boldsymbol{A}\left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1}(\boldsymbol{k} - \boldsymbol{A}\boldsymbol{\mu}) \\
\overline{\sigma}_i^2 &= \boldsymbol{e}_i^T\boldsymbol{\Sigma}\boldsymbol{e}_i - \boldsymbol{e}_i^T\boldsymbol{\Sigma}\boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T\right)^{-1}\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{e}_i
\end{aligned}$$

■

J.4. Proposition 9

Proposition 9 (Poisson Closed-form Expected Loss) *Let $\boldsymbol{z} = (z_1, \ldots, z_n)^T$, where $z_i \sim Poisson(\theta_i)$. Let $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ be the ground truth vector subject to the equality constraint $\sum_{i=1}^n y_i = k$ with $k \in \mathbb{N}^+$. Then it holds that*

i) when ℓ is L1 loss,

$$\begin{aligned} L(\boldsymbol{\theta}) = \sum_{i=1}^{n} & (k - \lfloor kp_i - d_i \rfloor) p_i Bin(\lfloor kp_i - d_i \rfloor; k, p_i) \\ & + \lfloor kp_i - d_i \rfloor (1 - p_i) Bin(\lfloor kp_i - d_i \rfloor; k, p_i) \\ & - 2d_i F(\lfloor kp_i - d_i \rfloor; k, p_i) + d_i; \end{aligned}$$

ii) when ℓ is L2 loss, $L(\boldsymbol{\theta}) = \sum_{i=1}^n kp_i(1-p_i) + k^2 p_i^2 - 2y_i kp_i + y_i^2$,
where Bin denotes the probability mass function (p.m.f.) of a binomial distribution and F denotes a regularized incomplete beta function. $d_i = kp_i - y_i$ and $p_i = \frac{\theta_i}{\sum_{j=1}^n \theta_j}$.

Proof Let $\boldsymbol{z} = (z_1, \ldots, z_n)^T$, where $z_i \sim Poisson(\theta_i)$. Let $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ be the ground truth vector subject to the equality constraint $\sum_{j=1}^n y_j = k$. We derive the closed-form expression for the L1 loss of $\boldsymbol{z}$ subject to the constraint $\sum_{j=1}^n z_j = k$.

$$\begin{aligned} L(\boldsymbol{\theta}) &= \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z} | \sum_{j=1}^n z_j = k)} [\| \boldsymbol{z} - \boldsymbol{y} \|_1] \\ &= \sum_{i=1}^{n} \mathbb{E}_{z_i \sim p(z_i | \sum_{j=1}^n z_j = k)} [|z_i - y_i|] \end{aligned}$$

Define $d_i = kp_i - y_i$, where $p_i = \frac{\theta_i}{\sum_{j=1}^n \theta_j}$. Then,

$$\begin{aligned} L(\boldsymbol{\theta}) &= \sum_{i=1}^{n} \mathbb{E}_{z_i \sim p(z_i | \sum_{j=1}^n z_j = k)} [|z_i - kp_i + d_i|] \\ &= \sum_{i=1}^{n} \sum_{all\ z_i} |z_i - kp_i + d_i| Binomial(z_i; k, p_i) \\ &= \sum_{z_i=0}^{\lfloor kp_i - d_i \rfloor} (-z_i + kp_i) Binomial(z_i; k, p_i) \\ &+ \sum_{\lfloor kp_i - d_i \rfloor}^{k} (z_i - kp_i) Binomial(z_i; k, p_i) \\ &- d_i \sum_{z_i=0}^{\lfloor kp_i - d_i \rfloor} Binomial(z_i; k, p_i) \\ &+ d_i \sum_{\lfloor kp_i - d_i \rfloor}^{k} Binomial(z_i; k, p_i) \end{aligned}$$

Consider the following lemma (Todhunter's Formula Diaconis and Zabell (1991))

Lemma 10 *For all integers* $0 \leq \alpha \leq \beta \leq n$,

$$\begin{aligned}&\sum_{x=\alpha}^{\beta}(x-np)Binomial(x;n,p)\\ =&\alpha(1-p)Binomial(\alpha;n,p)\\ -&(n-\beta)pBinomial(\beta;n,p)\end{aligned}$$

Then,

$$\begin{aligned}&\sum_{z_i=0}^{\lfloor kp_i-d_i\rfloor}(-z_i+kp_i)Binomial(z_i;k,p_i)\\ =&(k-\lfloor kp_i-d_i\rfloor)p_iBinomial(\lfloor kp_i-d_i\rfloor;k,p_i)\end{aligned}$$

and

$$\begin{aligned}&\sum_{\lfloor kp_i-d_i\rfloor}^{k}(z_i-kp_i)Binomial(z_i;k,p_i)\\ =&\lfloor kp_i-d_i\rfloor(1-p_i)Binomial(\lfloor kp_i-d_i\rfloor;k,p_i)\end{aligned}$$

Next, we notice that

$$\begin{aligned}&-d_i\sum_{z_i=0}^{\lfloor kp_i-d_i\rfloor}Binomial(z_i;k,p_i)\\ &+d_i\sum_{\lfloor kp_i-d_i\rfloor}^{k}Binomial(z_i;k,p_i)\\ =&-d_i\sum_{z_i=0}^{\lfloor kp_i-d_i\rfloor}Binomial(z_i;k,p_i)\\ &+d_i\left(1-\sum_{z_i=0}^{\lfloor kp_i-d_i\rfloor}Binomial(z_i;k,p_i)\right)\\ =&-2d_i\sum_{z_i=0}^{\lfloor kp_i-d_i\rfloor}Binomial(z_i;k,p_i)+d_i\end{aligned}$$

Define the regularized incomplete beta function as

$$F(x;n,p)=(n-x)\binom{n}{x}\int_0^{1-p}t^{n-x-1}(1-t)^x\,dt$$

Then,

$$\begin{aligned}&-2d_i\sum_{z_i=0}^{\lfloor kp_i-d_i\rfloor} Binomial(z_i;k,p_i)+d_i\\=&-2d_iF(\lfloor kp_i-d_i\rfloor;k,p_i)+d_i\end{aligned}$$

Thus, the closed-form expression for the L1 loss is

$$\begin{aligned}&L(\boldsymbol{\theta})\\=&\sum_{i=1}^{n}(k-\lfloor kp_i-d_i\rfloor)p_iBinomial(\lfloor kp_i-d_i\rfloor;k,p_i)\\&+\lfloor kp_i-d_i\rfloor(1-p_i)Binomial(\lfloor kp_i-d_i\rfloor;k,p_i)\\&-2d_iF(\lfloor kp_i-d_i\rfloor;k,p_i)+d_i\end{aligned}$$

We attempt to derive a closed-form solution for the L2 loss of $\boldsymbol{z}$ subject to the constraint $\sum_{j=1}^{n} z_j = k$.

$$\begin{aligned}L(\boldsymbol{\theta})&=\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{\theta}}(\boldsymbol{z}|\sum_i z_i=k)}[\parallel \boldsymbol{z}-\boldsymbol{b}\parallel_2^2]\\&=\sum_{i=1}^{n}\mathbb{E}_{z_i\sim p_{\boldsymbol{\theta}}(z_i|\sum_j z_j=k)}[z_i^2]\\&\quad-2\sum_{i=1}^{n}y_i\mathbb{E}_{z_i\sim p_{\boldsymbol{\theta}}(z_i|\sum_j z_j=k)}[z_i]+\sum_{i=1}^{n}y_i^2\end{aligned}$$

Since the conditional marginal distribution is a binomial distribution, it's second moment is given by

$$\begin{aligned}&\sum_{i=1}^{n}\mathbb{E}_{z_i\sim p_{\boldsymbol{\theta}}(z_i|\sum_j z_j=k)}[z_i^2]\\=&\sum_{i=1}^{n}k\left(\frac{\theta_i}{\sum_{j=1}^{n}\theta_j}\right)\left(\frac{\sum_{j=1}^{n}\theta_j-\theta_i}{\sum_{j=1}^{n}\theta_j}\right)+\left(\frac{k\theta_i}{\sum_{j=1}^{n}\theta_j}\right)^2\end{aligned}$$

It's first moment(mean) is given by

$$-2\sum_{i=1}^{n}y_i\mathbb{E}_{z_i\sim p_{\boldsymbol{\theta}}(z_i|\sum_j z_j=k)}[z_i]=-2k\sum_{i=1}^{n}y_i\left(\frac{\theta_i}{\sum_{j=1}^{n}\theta_j}\right)$$

Thus, we have

$$\begin{aligned}&\sum_{i=1}^{n}\mathbb{E}_{z_i\sim p_{\boldsymbol{\theta}}(z_i|\sum_j z_j=k)}[z_i^2]\\=&\sum_{i=1}^{n}k\left(\frac{\theta_i}{\sum_{j=1}^{n}\theta_j}\right)\left(\frac{\sum_{j=1}^{n}\theta_j-\theta_i}{\sum_{j=1}^{n}\theta_j}\right)+\left(\frac{k\theta_i}{\sum_{j=1}^{n}\theta_j}\right)^2\\&-2k\sum_{i=1}^{n}y_i\left(\frac{\theta_i}{\sum_{j=1}^{n}\theta_j}\right)+\sum_{i=1}^{n}y_i^2\end{aligned}$$

■

J.5. Proposition 11

Proposition 11 (Poisson Constrained Distribution) *Given* $\boldsymbol{z} = (z_1, \ldots, z_n)^T$ *with* $z_i \sim Poisson(\theta_i)$, *the constrained distribution* $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \sum_{j=1}^n z_n = k)$ *is equivalent to a multinomial distribution with parameter* k *and probabilities* $\frac{\theta_1}{\sum_{j=1}^n \theta_j}, \ldots, \frac{\theta_n}{\sum_{j=1}^n \theta_j}$.

Proof Let $\boldsymbol{z} = (z_1, \ldots, z_n)^T$, where $z_i \sim Poisson(\theta_i)$. We compute a closed-form solution for the conditional probability $p_{\boldsymbol{\theta}}\left(\boldsymbol{z} \mid \sum_{j=1}^n z_j = k\right)$.

$$p_{\boldsymbol{\theta}}\left((\boldsymbol{z}|\sum_{j=1}^n z_j = k\right) = \frac{p\left(\boldsymbol{z} \cap \sum z_i = k\right)}{p\left(\sum_{j=1}^n z_j = k\right)}$$

Let $Y = \sum_{j=1}^n z_j$. The denominator is the p.d.f. of Y evaluated at k. Since Y is a linear combination of independent Poisson random variables, we know $Y \sim Poisson(\sum_{j=1}^n \theta_j)$. Thus,

$$p\left(\sum_{j=1}^n z_j = k\right) = \frac{e^{-\sum_{j=1}^n \theta_j}\left(\sum_{j=1}^n \theta_j\right)^k}{k!}$$

Next, let's consider the numerator.

$$p(\boldsymbol{z} \cap \sum_{j=1}^n z_j = k) = \begin{cases} p(\boldsymbol{z}) & \sum_{j=1}^n z_j = k \\ 0 & \sum_{j=1}^n z_j \neq k \end{cases}$$

where $p(\boldsymbol{z}) = \prod_{i=1}^{n} f(z_i) = \prod_{i=1}^{n} \frac{e^{-\theta_i}\theta_i^{z_i}}{z_i!}$. Thus, our conditional distribution is given by

$$
\begin{aligned}
& p(\boldsymbol{z} | \sum_{j=1}^{n} z_j = k) \\
=& \begin{cases} \frac{\frac{e^{-\sum_{i=1}^{n}\theta_i}\prod_{i=1}^{n}\theta_i^{z_i}}{\prod_{i=1}^{n} z_i!}}{\frac{e^{-\sum_{i=1}^{n}\theta_i}\left(\sum_{i=1}^{n}\theta_i\right)^k}{k!}} & \sum_{j=1}^{n} z_j = k \\ 0 & \sum_{j=1}^{n} z_j \neq k \end{cases} \\
=& \begin{cases} \frac{k!\prod_{i=1}^{n}\theta_i^{z_i}}{(\sum_{i=1}^{n}\theta_i)^k \prod_{i=1}^{n} z_i!} & \sum_{j=1}^{n} z_j = k \\ 0 & \sum_{j=1}^{n} z_j \neq k \end{cases} \\
=& \begin{cases} \frac{1}{(\sum_{i=1}^{n}\theta_i)^k} \cdot \frac{k!}{\prod_{i=1}^{n} z_i!} \prod_{i=1}^{n}\theta_i^{z_i} & \sum_{j=1}^{n} z_j = k \\ 0 & \sum_{j=1}^{n} z_j \neq k \end{cases} \\
=& \begin{cases} \frac{k!}{\prod_{i=1}^{n} z_i!} \prod_{i=1}^{n} \left(\frac{\theta_i}{\sum_{j=1}^{n}\theta_j}\right)^{z_i} & \sum_{j=1}^{n} z_j = k \\ 0 & \sum_{j=1}^{n} z_j \neq k \end{cases} \\
=& f\left(\boldsymbol{z}; k, \frac{\theta_1}{\sum_{j=1}^{n}\theta_j}, \ldots, \frac{\theta_n}{\sum_{j=1}^{n}\theta_j}\right)
\end{aligned}
$$

where $f\left(\boldsymbol{z}; k, \frac{\theta_1}{\sum_{j=1}^{n}\theta_j}, \ldots, \frac{\theta_n}{\sum_{j=1}^{n}\theta_j}\right)$ is the probability mass function of a multinomial distribution with parameter k and $\frac{\theta_1}{\sum_{j=1}^{n}\theta_j}, \ldots, \frac{\theta_n}{\sum_{j=1}^{n}\theta_j}$. ■

J.6. Proposition 12

Proposition 12 (Poisson Conditional Marginal and Expectations) *Given $\boldsymbol{z} = (z_1, \ldots, z_n)^T$ with $z_i \sim Poisson(\theta_i)$, the conditional marginal $p_{\boldsymbol{\theta}}(z_i \mid \sum_{j=1}^{n} z_j = k)$ follows a binomial distribution with parameter k and probability $\frac{\theta_i}{\sum_{j=1}^{n}\theta_j}$. Further, its expectation is $\frac{k\theta_i}{\sum_{j=1}^{n}\theta_j}$.*

Proof Let $\boldsymbol{z} = (z_1, \ldots, z_n)^T$, where $z_i \sim Poisson(\theta_i)$. We compute a closed-form solution for the conditional marginal $p_{\boldsymbol{\theta}}(z_i \mid \sum_{j=1}^{n} z_n = k)$. Since the marginal of each variable of a multinomial distribution is a binomial distribution, then the conditional marginal is

$$
\begin{aligned}
& p\left(z_i \mid \sum_{j=1}^{n} z_j = k\right) \\
=& \binom{k}{z_i} \left(\frac{\theta_i}{\sum_{j=1}^{n}\theta_j}\right)^{z_i} \left(1 - \frac{\theta_i}{\sum_{j=1}^{n}\theta_j}\right)^{n-z_i}
\end{aligned}
$$

This is the probability mass function of a binomial distribution with parameter k and probability $\frac{\theta_i}{\sum_{j=1}^{n}\theta_j}$. ■

Appendix K. MOF Expected Loss NLL Explode

In this section, we provide a mathematical explanation for the negative log likelihood to explode for closed-form expected loss in the MOF experiment. The L1 loss function is given by

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \overline{\sigma}_i \sqrt{\frac{2}{\pi}} \exp\left(\frac{-(\overline{\mu}_i - y_i)^2}{2\overline{\sigma}_i^2}\right) + (\overline{\mu}_i - y_i)\, erf\left(\frac{\overline{\mu}_i - y_i}{\sqrt{2\overline{\sigma}_i^2}}\right)$$

with $\overline{\mu}_i := \boldsymbol{\mu}_i + \frac{\boldsymbol{\Sigma}_{i,i}}{\sum_{t=1}^{n} \boldsymbol{\Sigma}_{t,t}} \left(k - \sum_{j=1}^{n} \boldsymbol{\mu}_j\right)$ and $\overline{\sigma}_i^2 := \boldsymbol{\Sigma}_{i,i} - \frac{(\boldsymbol{\Sigma}_{i,i})^2}{\sum_{t=1}^{n} \boldsymbol{\Sigma}_{t,t}}$. Define a constant c such that $0 < c < 1$. Notice that if we scale the unconstrained variance $\boldsymbol{\Sigma}_{i,i}$ by c, $\overline{\mu}_{i,\text{scaled}} = \overline{\mu}_i$ and $\overline{\sigma}^2_{i,\text{scaled}} = c\overline{\sigma}_i^2$.

$$\begin{aligned}
&\lim_{c\to 0} \overline{\sigma}_{i,\text{scaled}} \sqrt{\frac{2}{\pi}} \exp\left(\frac{-(\overline{\mu}_{i,\text{scaled}} - y_i)^2}{2\overline{\sigma}^2_{i,\text{scaled}}}\right) \\
=&\lim_{c\to 0} c\overline{\sigma}_i \sqrt{\frac{2}{\pi}} \exp\left(\frac{1}{c}\frac{-(\overline{\mu}_i - y_i)^2}{2\overline{\sigma}_i^2}\right) \\
=&0
\end{aligned}$$

Also,

$$\begin{aligned}
&\lim_{c\to 0} (\overline{\mu}_{i,\text{scaled}} - y_i)\, erf\left(\frac{\overline{\mu}_{i,\text{scaled}} - y_i}{\sqrt{2\overline{\sigma}^2_{i,\text{scaled}}}}\right) \\
=&\lim_{c\to 0} (\overline{\mu}_i - y_i)\, erf\left(\frac{1}{\sqrt{c}}\frac{\overline{\mu}_i - y_i}{\sqrt{2\overline{\sigma}_i^2}}\right) \\
=&|\overline{\mu}_i - y_i|
\end{aligned}$$

As the scaling factor decreases, the expected loss converges, which shows that the expected loss favors variance with smaller magnitude. However, MAE is only associated with the constrained mean, which is invariant to scaling of the variance. Extremely small variance causes the constrained distribution to approach the shape of a dirac delta function, causing the NLL to explode.

Appendix L. Scalability with respect to Dimension and Constraints

We first present the results of exact sampling, computing closed-form loss, and gradient estimator with respect to varying dimensions. The number of constraints is half of the dimension (n). All results are averaged over 100 runs and conducted on a Nvidia A800 80GB GPU. All numbers are measured in seconds. We highlight that even though increasing dimension slows down the algorithm, the additional computational cost can be

mitigated by choosing a larger batch size. For example, the additional time taken to run with `batch_size`=16 and `batch_size`=32 is almost negligible for $n = 32$, 64, and 128. In the challenging $n = 2048$ case, using `batch_size`=32 only takes 1.5 times more time to run compared to `batch_size`=16. Larger batch sizes can significantly reduce computation time per sample thanks to the parallelization of our method.

Table 11: Runtime (in seconds) for varying dimensions and batch sizes.

	16	32	64	128	256
$n = 32$	0.00249	0.00250	0.00252	0.00253	0.00255
$n = 64$	0.00348	0.00348	0.00348	0.00352	0.00366
$n = 128$	0.00467	0.00473	0.00479	0.00483	0.00498
$n = 256$	0.00760	0.00769	0.00806	0.00916	0.01052
$n = 512$	0.00760	0.01770	0.02159	0.02921	0.04500
$n = 1024$	0.04843	0.06266	0.09138	0.13943	0.23699
$n = 2048$	0.20515	0.31654	0.46224	0.78970	1.24126

Algorithm 2 Scalability with Varying Dimensions

for each $n \in ls\_n$ **do**
  $num\_constraints \leftarrow n/2$
  $Total\_time \leftarrow 0$
  for $i \leftarrow 1$ **to** $repeat\_num$ **do**
    Generate unconstrained parameters and constraints
    Conduct exact sampling
    Compute closed-form loss
    Compute gradient estimator
    $Total\_time \leftarrow Total\_time + \text{time}$
  end for
  $Total\_time \leftarrow Total\_time/repeat\_num$
  print($Total\_time$)
end for

We next investigate the impact of the number of constraints (`num_k`). We conduct experiments on the challenging $n = 2048$ setting. Similar to the findings in scaling the number of dimensions, using larger batch sizes significantly reduces computation time per sample. Additionally, we also found that increasing the number of constraints does not significantly increase the computational time. For example, under `batch_size` = 64, increasing the number of constraints from 256 to 512 only increases computational time by 32%.

Table 12: Runtime (in seconds) for varying number of constraints and batch sizes at $n = 2048$.

	16	32	64	128	256
num_k = 32	0.05109	0.09899	0.19531	0.29093	0.47706
num_k = 64	0.05343	0.10259	0.20241	0.30316	0.48106
num_k = 128	0.05894	0.11060	0.21666	0.32456	0.50057
num_k = 256	0.07336	0.13211	0.24871	0.38556	0.52001
num_k = 512	0.10885	0.18119	0.32935	0.52796	0.84375
num_k = 1024	0.20515	0.31654	0.46224	0.78970	1.24126

Appendix M. Hardware and Software Specification

We implement our model in PyTorch. All experiments are run on servers/workstations with the following configuration:

- 32 CPUs, 128G Mem, 4 × NVIDIA A5000 GPUs. Ubuntu 22.04.4
- 128 CPUs, 480G Mem, 8 × NVIDIA RTX 4090 GPUs. Ubuntu 22.04
- 48 CPUs, 200G Mem, 8 × NVIDIA V100 GPUs. Ubuntu 22.04

Appendix N. Additional Experiment on Comparison of Gradient Estimators

We vary the gradient dimension in the Comparison of Gradient Estimators.

We compute the bias on four different dimension $10, 30, 70, 100$. Results are presented in Figure 10. The number of samples to estimate the bias is fixed to 10^4 for all dimensions.

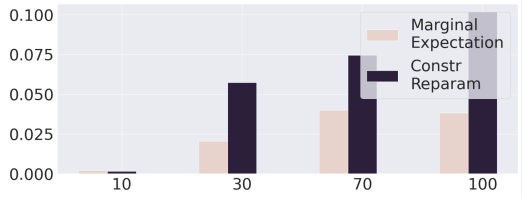

Figure 10: Bias Comparison of Marginal Expectation and Constrained Reparametrization with Varying Dimension

The results for Variance and Average Error are presented below:

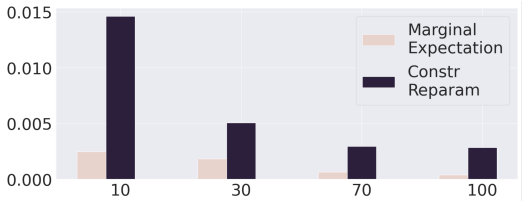

Figure 11: Variance Comparison of Marginal Expectation and Constrained Reparametrization with Varying Dimension

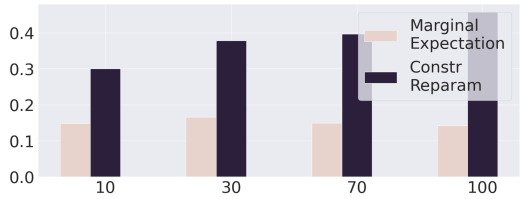

Figure 12: Average Error Comparison of Marginal Expectation and Constrained Reparametrization with Varying Dimension