SHAPING THE LEARNING GRADIENT DISTRIBUTION WITH THERMODYNAMIC INTEGRATION

CUED IIB REPORT

Prithvi Raj
Department of Engineering
University of Cambridge
pr478@cam.ac.uk

Mark Girolami
Department of Engineering
University of Cambridge
mag92@cam.ac.uk

June 12, 2024

ABSTRACT

The performances of deep generative models depend on the distributional characteristics of their learning gradients. Despite this, their exact influence remains poorly understood, and investigations into the topic are bounded by our limited ability to shape the learning gradient distribution. To address this, we propose leveraging Thermodynamic Integration as a means of robustly controlling the learning gradient variance. The method is subsequently applied to investigate the relationship between learning gradient variance and the fidelity of images generated by the latent space energy-based prior model, introduced by Pang et al. [12], to reveal striking polynomial relationships between image fidelity and learning gradient variance.

1 Introduction

Deep generative models belong to a category of machine learning algorithms characterised as neural networks for generating new data samples, such as images or text. Such models undergo training primarily by minimising a loss function, denoted as $\mathcal{L}(\theta, \mathbf{x})$, achieved through iterative adjustments of the neural network parameters, represented as θ. The form of these adjustments are evaluated through gradient-based optimisation techniques, such as stochastic gradient descent (SGD) [13] and Adam optimisation [8].

Importantly, these methods require the learning gradient, $\nabla_\theta \mathcal{L}(\theta, \mathbf{x})$, i.e. the gradient of the loss with respect to the parameters, evaluated at a training sample $\mathbf{x}$. However, due to the stochastic approximation methods required to evaluate $\mathcal{L}(\theta, \mathbf{x})$, the learning gradient forms a non-deterministic probability distribution.

In agreement with the previous work of [5], we therefore argue that adopting a distributional perspective on gradient-based optimisation and exploring the characteristics of $\nabla_\theta \mathcal{L}(\theta, \mathbf{x})$ are crucial steps toward enhancing optimisation speed and the model's ability to generalise beyond the training dataset and produce high fidelity, diverse samples. However, exerting explicit control over the shape of this distribution presents a persistent obstacle that has prevented researchers from fully grasping its exact influence in the optimisation of deep neural networks.

For example, Gradient Clustering (GC) is presented in [5] as a means of minimising gradient variance. However, the work does not conclusively show that convergence speed or accuracy is improved by minimising gradient noise via GC. It also does not present Gradient Clustering as a means of arbitrarily controlling gradient variance, which is desirable since larger learning gradient variances often lead to improved generative capacity, as shown later in this study. Minimising variance might be justifiable if $\nabla_\theta \mathcal{L}(\theta, \mathbf{x})$ were a value, but it is instead a distribution.

Another study showcased in [2] demonstrated that employing larger mini-batch sizes, resulting in smaller gradient noise during Generative Adversarial Network (GAN) training, substantially enhanced the quality of generated samples and improved training stability. This suggests that smaller variances in $\nabla_\theta \mathcal{L}(\theta, \mathbf{x})$ are favourable, which is generally not consistent across different studies. Additionally, this may suggest that batch size emerges as a promising method to control the learning gradient variance. However, batching is limited by computational memory requirements, and is not a consistent or predictable means of shaping the learning gradient distribution, as demonstrated later in this study.

Moreover, the previous work of [11] revealed that injecting additional gradient noise into very deep generative networks enhanced model performance by addressing overfitting issues and reducing training loss. However, despite being easy to implement, injecting noise is not a robust and controllable means of shaping the distribution of $\mathcal{L}(\theta, \mathbf{x})$, and too much gradient noise results in comparatively worse generative capacity, as shown later in this study.

Collectively, these prior studies demonstrate the limited comprehension among machine learning practitioners regarding the distributional characteristics of $\nabla_\theta \mathcal{L}(\theta, \mathbf{x})$, despite its pivotal role in optimising the performance of deep generative models. Hence, we demonstrate Thermodynamic Integration as a method to effectively shape this distribution by directly parameterising its variance, $\mathrm{Var}_\theta\left[\nabla_\theta \mathcal{L}(\theta, \mathbf{x})\right]$, aiming for its adoption in similar investigations concerning gradient-based learning.

To comprehensively explore the capabilities of Thermodynamic Integration, the generative capacity of the latent space energy-based prior model proposed in [12] was investigated using Thermodynamic Integration.

2 The latent space energy-based prior model

First, we present the particular model under scrutiny. The complete derivations of the underlying formulations are detailed in [12], so they are not repeated here. The latent space energy-based prior model proposed in [12] leverages both latent space representations and Markov chain Monte Carlo (MCMC) techniques to generate high-fidelity images, despite tolerating a notable amount of statistical flexibility in its MCMC sampling procedure.

This statistical flexibility introduces a degree of variance into the latent variable estimates produced by MCMC sampling, as well as its overall marginal likelihood evaluation. This marks the latent space energy-based prior model as a prime candidate for investigating the influence of learning gradient variance using Thermodynamic Integration.

2.1 Defining the networks

The novelty of this model was its inclusion of an energy-based model (EBM) to learn the latent representation. After first sampling the latent variable $\mathbf{z}$ from a simple prior distribution, e.g. $\pi_0(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \sigma_0 \mathbf{I})$, (which can be realised easily using standard random number generators), the sample was then passed through an EBM to exponentially-tilt the latent distribution producing a parameterised latent prior:

$$p_\alpha(\mathbf{z}) = \frac{1}{Z_\alpha} \exp(f_\alpha(\mathbf{z})) \cdot \pi_0(\mathbf{z}) \tag{1}$$

Here, α represents the EBM's learnable parameters, $f_\alpha(\mathbf{z})$ denotes the EBM's output when fed an input $\mathbf{z} \sim \pi_0(\mathbf{z})$, and Z_α represents the normalisation constant included to ensure the validity of the probability distribution.

A sample from this parametrised prior, $p_\alpha(\mathbf{z})$, subsequently serves as an input to the generator network (GEN), which maps the latent representation back to the data space:

$$\tilde{\mathbf{x}} = g_\beta(\mathbf{z}) + \epsilon \tag{2}$$

Here, β denotes the learnable parameters of the GEN, $\tilde{\mathbf{x}}$ signifies the generated data, $g_\beta(\mathbf{z})$ represents the output of the GEN, and $\epsilon \sim \mathcal{N}(0, \sigma_l \mathbf{I})$ indicates the standard noise term of the top-down network of the VAE class of generative models, (which is the family of networks that the GEN can be characterised into).

The effective parameterisation of the latent prior distribution in Eq. 1 enables the iterative refinement of the latent representation based on observations from the dataset. This is essentially achieved through training the EBM network to shape the prior into a distribution that loosely satisfies $p_\alpha(\mathbf{z}) \approx p_\theta(\mathbf{z}|\mathbf{x})$, where θ is used here to define the set containing all model parameters, i.e. $\alpha, \beta \in \theta$.

The posterior distribution, $p_\theta(\mathbf{z}|\mathbf{x})$, can be considered a data-informed latent distribution. It is a distribution of $\mathbf{z}$ conditioned on observations from the real dataset used to train the model, $\mathbf{x}$. Therefore, the prior model in 1 can be interpreted as an energy-based, data-informed correction or exponential tilting of the original prior distribution $\pi_0(\mathbf{z})$, informed by features of $\mathbf{x}$. This allows the EBM to exert control over the latent distribution and shape it accordingly, resulting in a more refined latent representation that is better suited to capturing essential features in the latent space.

When creating a new generation of $\tilde{\mathbf{x}}$, the latent variable $\mathbf{z}$ is then sampled directly from $p_\alpha(\mathbf{z})$ and passed through Eq. 2. It is expected that after training the EBM, $p_\alpha(\mathbf{z})$ is similar enough to $p_\theta(\mathbf{z}|\mathbf{x})$ that the process is akin to sampling from $p_\theta(\mathbf{z}|\mathbf{x})$ itself and passing the result through Eq. 2, which is not feasible without having new reference data $\mathbf{x}$, required to evaluate samples from $p_\theta(\mathbf{z}|\mathbf{x})$.

2.2 Training

The complete model, comprising both networks, is then trained using the maximum likelihood approach:

$$\begin{aligned} \hat{\theta} &= \underset{\theta}{\operatorname{argmin}} \; -p_\theta(\mathbf{x}) \\ &= \underset{\theta}{\operatorname{argmin}} \; \mathcal{L}(\theta, \mathbf{x}) \end{aligned} \tag{3}$$

Here, $\mathcal{L}(\theta, \mathbf{x})$ denotes the loss function. Training the two neural networks involves updating the networks' parameters in the direction of the gradient of the loss function with respect to those parameters:

$$\nabla_\theta \mathcal{L}(\theta, \mathbf{x}) = -\nabla_\theta \log(p_\theta(\mathbf{x})) \tag{4}$$

This can be separated into two terms corresponding to each set of model parameters, by first rewriting the likelihood as a marginalisation over the entire latent distribution:

$$\begin{aligned}\nabla_\theta \log(p_\theta(\mathbf{x})) &= \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\left[\nabla_\theta \log(p_\theta(\mathbf{x},\mathbf{z}))\right] \\ &= \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\left[\nabla_\theta \log(p_\alpha(\mathbf{z})p_\beta(\mathbf{x}|\mathbf{z}))\right] \\ &= \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\left[\nabla_\alpha \log(p_\alpha(\mathbf{z}))\right] \\ &+ \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\left[\nabla_\beta \log(p_\beta(\mathbf{x}|\mathbf{z}))\right] \end{aligned} \tag{5}$$

The term corresponding to the EBM model can be expanded as

$$\nabla_\alpha \log p_\alpha(\mathbf{z}) = \nabla_\alpha f_\alpha(\mathbf{z}) - \mathbb{E}_{p_\alpha(\mathbf{z})}\left[\nabla_\alpha f_\alpha(\mathbf{z})\right] \tag{6}$$

With this in mind, the learning gradients required to update each neural network's set of parameters are therefore:

$$\begin{aligned}\delta_\alpha(\mathbf{x}) &= \nabla_\alpha \log p_\theta(\mathbf{x}) \\ &= \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\left[\nabla_\alpha f_\alpha(\mathbf{z})\right] - \mathbb{E}_{p_\alpha(\mathbf{z})}\left[\nabla_\alpha f_\alpha(\mathbf{z})\right] \end{aligned} \tag{7}$$

$$\delta_\beta(\mathbf{x}) = \nabla_\beta \log p_\theta(\mathbf{x}) = \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\left[\nabla_\beta \log p_\beta(\mathbf{x}|\mathbf{z})\right] \tag{8}$$

In Eq. 7, α is updated based on the difference between the EBM's output using $\mathbf{z}$ sampled from two different probability distributions, $p_\theta(\mathbf{z}|\mathbf{x})$ and $p_\alpha(\mathbf{z})$. Therefore, the EBM's update step serves to tune the exponential-tilting of Eq. 1 to shape $p_\alpha(\mathbf{z})$ into a distribution that is more representative of the fundamental features present in $\mathbf{x}$, despite the distribution's lack of dependence on $\mathbf{x}$.

In the context of image generation, the likelihood term in Eq. 8 is given as:.

$$\log p_\beta(\mathbf{x}|\mathbf{z}) = -\frac{\|\mathbf{x} - g_\beta(\mathbf{z})\|^2}{2\sigma_l^2} + \text{constant} \tag{9}$$

2.3 Latent space sampling with Langevin dynamics

For the expectations in Eq. 7 and 8, MCMC sampling is used to estimate samples of $\mathbf{z}$ from $p_\alpha(\mathbf{z})$ and $p_\theta(\mathbf{z}|\mathbf{x})$. In [12], the particular MCMC technique used was unadjusted Langevin Sampling. Given a target distribution, $q(\mathbf{z})$, the MCMC update step is computed as:

$$\begin{aligned}\mathbf{z}_{k+1} &= \mathbf{z}_k + s\nabla_\mathbf{z} \log q(\mathbf{z}_k) + \sqrt{2s}\epsilon_\mathbf{k}, \\ \text{where} \quad k &= 1, \ldots, K_\mathbf{z}\end{aligned} \tag{10}$$

Here, $K_\mathbf{z}$ denotes the total number of steps that the MCMC sampling loop is conducted for, k indexes the incremental step of the Langevin algorithm, s is a small step size, and $\epsilon_\mathbf{k} \sim \mathcal{N}(\mathbf{0}, \mathbf{I_d})$ is Gaussian white noise. The starting sample, $\mathbf{z}_0$, is initialised from a predefined distribution, $\mathbf{z}_0 \sim q_0(\mathbf{z})$, and the resulting $\mathbf{z}_{K_\mathbf{z}}$ is the final sample from the estimated target distribution, $q(\mathbf{z})$.

In the case of sampling from $p_\alpha(\mathbf{z})$:

$$\begin{aligned}q(\mathbf{z}) &= p_\alpha(\mathbf{z}), \\ q_0(\mathbf{z}) &= \pi_0(\mathbf{z}), \\ \nabla_\mathbf{z} \log q(\mathbf{z}_k) &= \nabla_\mathbf{z} \log p_\alpha(\mathbf{z}) = \nabla_\mathbf{z} f_\alpha(\mathbf{z}) - \frac{\mathbf{z}}{\sigma_0^2}\end{aligned} \tag{11}$$

In the case of sampling from $p_\theta(\mathbf{z}|\mathbf{x})$:

$$\begin{aligned}q(\mathbf{z}) &= p_\theta(\mathbf{z}|\mathbf{x}), \\ q_0(\mathbf{z}) &= \pi_0(\mathbf{z}), \\ \nabla_\mathbf{z} \log q(\mathbf{z}_k) &= \nabla_\mathbf{z} \log p_\theta(\mathbf{z}|\mathbf{x}) \\ &= \nabla_\mathbf{z} \log p_\beta(\mathbf{x}|\mathbf{z}) + \nabla_\mathbf{z} \log p_\alpha(\mathbf{z}) \\ &= -\nabla_\mathbf{z} \frac{\|\mathbf{x} - g_\beta(\mathbf{z})\|^2}{2\sigma_l^2} + \nabla_\mathbf{z} f_\alpha(\mathbf{z}) - \frac{\mathbf{z}}{\sigma_0^2}\end{aligned} \tag{12}$$

2.4 Sources of variance in the MCMC methods

Guaranteeing full convergence to an intricate target distribution in MCMC sampling typically requires an infinite number of steps with infinitesimal step sizes, which is infeasible. Instead, the Langevin sampling loop used in the previous work of [12] adopts a statistically flexible approach in its implementation. This introduces noticeable, unintended variance into the value that the final sample, $(\mathbf{z}_K)$, converges on, in addition to the Monte Carlo error later elucidated in Eq. 14.

Firstly, The total MCMC procedure was conducted for a small number of iterations, e.g. $K_{\mathbf{z}|\mathbf{x}} = 20$ in Eq. 10. This limited iteration count diminishes the procedure's capacity to thoroughly explore the target distribution. Consequently, the resulting final sample may have failed to faithfully represent the target distribution. The samples are less likely to converge towards any specific mode, and may instead be spread across the distribution's modes, introducing more variance into the final sample.

Additionally, the Langevin sampling loop in Eq. 10 operated without error correction. The procedure did not include any mechanism to reject samples based off the relative probability that the proposal state takes, (as otherwise implemented in Metropolis-adjusted Langevin sampling). Without any rejection mechanism, there is no means of correcting or reducing errors, which may even accumulate during each iteration. This similarly introduced a degree of variance into the value that the final sample adopts.

Furthermore, the marginal likelihood evaluation described in Eq. 5 required the calculation of expectations. This requires marginalising or integrating over the entire latent

space, which is an infeasible computation. Instead, the loss was approximated in [12] through the Monte Carlo estimator of the mean:

$$\begin{aligned} p_\theta(\mathbf{x}) &= \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\left[p_\theta(\mathbf{x},\mathbf{z})\right] \\ &= \int p_\theta(\mathbf{z}|\mathbf{x}) p_\theta(\mathbf{x},\mathbf{z})\, d\mathbf{z} \\ &\approx \frac{1}{N}\sum_{i=1}^{N} p_\theta(\mathbf{x},\mathbf{z}^{(i)}) \\ &= \overline{\rho}(N) \end{aligned} \tag{13}$$

Here, $\mathbf{z}^{(i)}$ represents the ith posterior sample taken from the set $\left\{\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots, \mathbf{z}^{(N)}\right\}$, with N indicating the total number of samples in the set. However, it can be shown that the Monte Carlo error in this estimator scales with the inverse root of the sample size, N [9]:

$$\sqrt{\mathbb{E}\left[\overline{\rho}(N)^2\right] - \mathbb{E}\left[\overline{\rho}(N)\right]^2} \propto \frac{1}{\sqrt{N}} \tag{14}$$

In the previous work of [12], the sample sizes used in any MCMC estimators were defined through batching and the size of the latent representation. As such, these estimators emerged as significant sources of variance that manifested in the log-marginal likelihood evaluation and therefore the learning gradient. However, despite these sources of variance, the model demonstrated a large capacity for generating high fidelity images when trained on the SVHN, CIFAR-10, and CelebA datasets.

This prompts the study's inquiry regarding the significance of learning gradient variance within latent space modeling for image generation. Thermodynamic Integration emerges as a well-suited means of creating the range of variances required to answer these questions, without requiring additional computational resources.

3 Thermodynamic Integration

Trying to reduce the MCMC estimator's variance through batch or sample size alone proves challenging. Firstly, increasing sample size to reduce MCMC error is limited by computational memory requirements. Secondly, the effect does not remain consistent across repetitions.

Instead, we apply Thermodynamic Integration towards obtaining the marginal likelihood as presented in [3]. For the same sample size, Thermodynamic Integration can be structured to parameterise the learning gradient variance, and achieve its explicit control without requiring more storage. Instead, its implementation requires more compute time, which is influenced by N_t in Eq. 18.

3.1 The thermodynamic integral

In place of the formulation and approximation outlined in eqs. 5 and 13, this method exploits the following representation of the logarithm of the marginal likelihood:

$$\log p_\theta(\mathbf{x}) = \int_0^1 \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}|\mathbf{z})\right]\, dt \tag{15}$$

Here, $p_\theta(\mathbf{z}|\mathbf{x},t)$ has been introduced as the power posterior, which can be considered a representation of the original posterior distribution, tempered by a temperature parameter, t. This is presented in [6] and organised for our purposes as follows:

$$\begin{aligned} p_\theta(\mathbf{z}|\mathbf{x},t) &= \frac{p_\beta(\mathbf{x}|\mathbf{z})^t p_\alpha(\mathbf{z})}{\mathcal{Z}(\mathbf{x}|t)}, \\ \text{where } \mathcal{Z}(\mathbf{x}|t) &= \int p_\beta(\mathbf{x}|\mathbf{z})^t p_\alpha(\mathbf{z}) d\mathbf{z} \end{aligned} \tag{16}$$

Given the formulation in Eq. 17, setting $t = 0$ results in the posterior distribution assuming the form of the prior distribution. As t increments, the posterior gradually adopts a shape more closely resembling the true Bayesian posterior distribution. At $t = 1$, the posterior converges to the true Bayesian posterior distribution.

The tempered posterior also alters the update step in Eq. 12, given that the likelihood term has been raised to the power of t:

$$\begin{aligned} \nabla_{\mathbf{z}} \log q(\mathbf{z}_k) &= \nabla_{\mathbf{z}} \log p_\theta(\mathbf{z}|\mathbf{x},t) \\ &\propto \nabla_{\mathbf{z}} \log p_\beta(\mathbf{x}|\mathbf{z})^t + \nabla_{\mathbf{z}} \log p_\alpha(\mathbf{z}) \\ &\propto -\nabla_{\mathbf{z}} \frac{t \cdot \|\mathbf{x} - g_\beta(\mathbf{z})\|^2}{2\sigma_l^2} + \nabla_{\mathbf{z}} f_\alpha(\mathbf{z}) - \frac{\mathbf{z}}{\sigma_0^2} \end{aligned} \tag{17}$$

Setting $t = 0$ results in the MCMC sampling procedure yielding an estimate from the prior distribution, while increasing t progressively facilitates exploration of the true posterior distribution. Therefore, aside from addressing error in the Monte Carlo estimator, using the thermodynamic integral also allows for more nuanced MCMC exploration of the posterior distribution, which mitigates the variances associated with the unadjusted Langevin algorithm discussed in Section 2.4.

3.2 Discretisation of the thermodynamic integral

The exact calculation of the expectation in Eq. 15 remains practically infeasible, so following the work of [3], the temperature schedule is instead discretised as:

$$\{t_1, t_2, \ldots t_{N_\mathrm{t}}\} \text{ for } t_i \in [0,1] \tag{18}$$

Here, t_i denotes the tempering at the ith index of the schedule, and N_t is the number of temperatures. As such, the the evaluation in Eq. 15 is approximated as:

$$\log p_\theta(\mathbf{x}) = \frac{1}{2}\sum_i \Delta t_i (E_{i-1} + E_i) + \frac{1}{2}\sum_i D_{\mathrm{KL}}(p_{i-1}||p_i) + D_{\mathrm{KL}}(p_i||p_{i-1}) \tag{19}$$

Where:

$$\Delta t_i = t_i - t_{i-1}$$
$$E_i = \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t_i)}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$$
$$D_{\mathrm{KL}}(p_{i-1}||p_i) = \int p_\theta(\mathbf{z}|\mathbf{x},t_{i-1}) \frac{p_\theta(\mathbf{z}|\mathbf{x},t_{i-1})}{p_\theta(\mathbf{z}|\mathbf{x},t_i)} dz$$

This approximation is equivalent to using the trapezium rule for numerical integration. Its full derivation is presented in [3], and is computationally realised using Monte Carlo estimates for each E_i, which remains unavoidable and serves as a source of error that is again limited by the memory requirements associated with large sample sizes.

The second source of error arises from the Kullback-Leibler Divergence term, $D_{\mathrm{KL}}(p_{i-1}||p_i)$ in Eq. 19, an integration that requires approximation. The error in this approximation is bound by the number of partitions incorporated in the temperature schedule, N_t. The constraint on N_t stems from computation time rather than memory resources. A higher value of N_t scales the number of evaluations needed to compute Eq. 19, thereby elongating the time required for the marginal likelihood calculation.

Noting that the power posterior in Eq. 17 is Gaussian distributed, the KL divergence term used for Eq. 19 is computationally realised using the following analytic expression from [3]:

$$D_{\mathrm{KL}}(p_{i-1}||p_i) = \frac{1}{2}\Big(\ln\left(\frac{\det(\Sigma_i)}{\det(\Sigma_{i-1})}\right) + \mathrm{tr}(\Sigma_i^{-1}\Sigma_{i-1}) + (\mu_i - \mu_{i-1})^T\Sigma_i^{-1}(\mu_i - \mu_{i-1}) - d\Big) \tag{20}$$

Here, Σ_i and Σ_{i-1} are the covariance matrices of the distributions p_i and p_{i-1} respectively; μ_i and μ_{i-1} are the means, and d is the dimensionality, representing the number of dimensions in the multivariate space.

3.3 Temperature scheduling can control variance

We opt to use a power law formation to parameterise the schedule using a temperature power term, denoted as p:

$$t_i = \left(\frac{i}{N_t}\right)^p \quad \forall i \in \{1, \ldots, N_t\} \tag{21}$$

Given that the bounds of the continuous integral in Eq. 15 are maintained at $t = 0$ and $t = 1$, the choice of temperature schedule does not affect the validity of the approximation. Figs. 2 shows the impact of adjusting p. A higher p amplifies the influence of posteriors tempered by lower t values, indicating a greater emphasis on the simpler prior distribution and reduced exploration of the true posterior distribution.

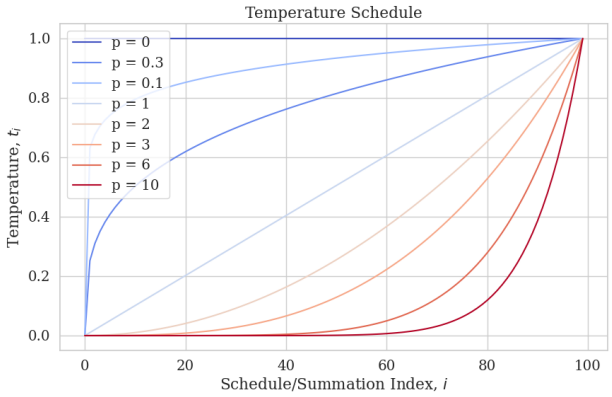
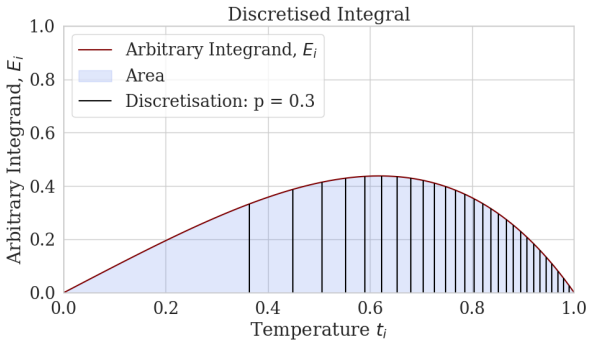
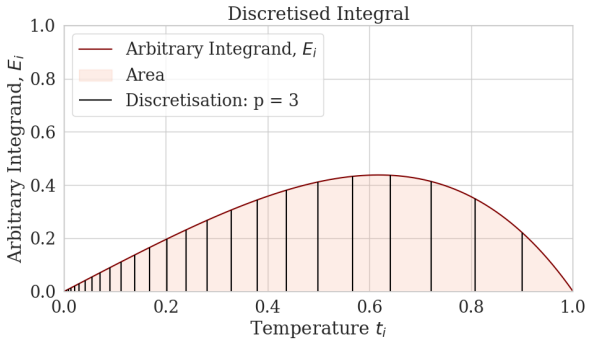


Figure 1: Schedules parameterised by p in Eq. 21. This reflects how the temperature, t in Eq. 17, increments with respect to the index of summation in Eq. 19.



(a) $p = 0.3$ schedule has more bins clustered towards $t = 1$.



(b) $p = 3$ schedule has more bins clustered towards $t = 0$.

Figure 2: Visualisation of different temperature schedules. These plots depict how the shape of the schedule affects the trapezoidal integral approximation. Evaluation points can be purposefully skewed towards a particular bound of the thermodynamic integral.

Given that the Bayesian posterior distribution is more complex in form than the prior distribution, a temperature schedule with partitions favouring lower temperatures, i.e., $p > 1$, is anticipated to elevate global exploration of $p_\theta(\mathbf{z}|\mathbf{x})$, when compared against a temperature schedule characterised by partitions favouring higher temperatures i.e., $0 < p \leq 1$, which is expected to elevate exploitation of its local form. Therefore, the choice of temperature schedule explicitly tips the balance between global exploration and local exploitation of the posterior sampling landscape.

This enables the precise control of learning gradient variance necessary for the proposed study. The variance in $\nabla_\theta \log(p_\theta(\mathbf{x}))$ can be regulated through Thermodynamic Integration, given its dependence on $p_\theta(\mathbf{z}|\mathbf{x}, t)$ in Eq. 19. Variances in samples from tempered posterior distributions manifest in downstream computations of $\nabla_\theta \log(p_\theta(\mathbf{x}))$.

4 Experiment

To explore the influence of learning gradient variance in the latent space energy-based prior model, we must assess how its training dynamics and generative capabilities vary with respect to the gradient variance. In our method, we achieve variation in this gradient noise by adjusting either the batch size used or by specifying the value of p in equation 21. Both have been included to cement the potential of Thermodynamic Integration over batching.

All models were trained at fixed complexities, which were reduced from those detailed in [12], until approximate convergence was achieved in both image quality and training gradient variance, (which required 50 epochs). Firstly, the unaltered (vanilla) model was analysed across different batch sizes, $\{25, 50, 75, 150\}$, to assess the effect of batch size on training gradient variance and image fidelity. Subsequently, an altered model incorporating Thermodynamic Integration with varying values of p in Eq. 21 was investigated, maintaining a fixed batch size of 75.

These training loops were repeated five times for each model to ensure experimental rigor, yielding five distinct readings to provide an indication of the uncertainty/robustness of the final gradient variances and image qualities produced by each model. The experiment was replicated using two datasets, CelebA and CIFAR-10, to confirm the consistency of the observed relationships across different datasets. Both are well-established in current literature and are adopted here to remain consistent.

During the training procedures, we introduced $\overline{KID}_\infty$ as a metric to assess image fidelity. In contrast to the commonly used Fréchet Inception Distance (FID) metric [7], we found Kernel Inception Distance (KID) [1] to be a more reliable measure of image fidelity. Moreover, to mitigate potential biases inherent in these metrics, we adopted a methodology inspired by the work of [4]. This involved collecting KID readings across various sample sizes and using linear regression to extrapolate to an infinitely-sized sample set. This approach yields an unbiased estimator of the metric, thereby improving the robustness of our perceptual quality evaluation process. We denote this unbiased estimator as $\overline{KID}_\infty$. Lower $\overline{KID}_\infty$ values correspond to a higher quality set of images.

Following this, the experiment was extended to examine the impact of:

- The amount of temperature discretisation, denoted as N_t in Equation 18
- The number of MCMC iterations required to sample from $p_\theta(\mathbf{z}|\mathbf{x}, t)$, denoted as $K_{\mathbf{z}|\mathbf{x},t}$ in Eq. 11
- The weighting applied to the KL divergence terms of Eq. 19, denoted as η in Eq. 22.

This extension focused solely on the $p = 0.1$ model. Despite demonstrating a comparable learning gradient variance to the vanilla models, this model generated lower fidelity images, attributed to its convergence to a local optima in the loss landscape.

In the model incorporating Thermodynamic Integration, $\nabla_\alpha \mathcal{L}(\theta, \mathbf{x})$ is solely tracked through the MCMC procedure used to evaluate samples from $p_\theta(\mathbf{z}|\mathbf{x}, t)$. This tracking is significantly reduced with $p = 0.1$, which corresponds to a significant decrease in the influence of the prior, attributed to its specific temperature schedule, (see Fig. 2). This exacerbates the altered model's lack of prior matching, previously apparent through Eq. 7, and the exclusive dependence on Langevin sampling to track the gradients required to update α.

Relying solely on the MCMC sampling procedure for learning gradients is especially damaging to the EBM's learning capacity. Firstly, initial MCMC steps are taken into account, which are noisy and potentially biased. Secondly, if the MCMC sampler gets stuck in a local mode and fails to explore the other modes effectively, the gradients computed from that sample will be biased towards that particular mode. Consequently, the model may suffer from mode-dropping or mode-collapse issues.

Therefore, $K_{\mathbf{z}|\mathbf{x},t}$ was studied in more detail. Increasing $K_{\mathbf{z}|\mathbf{x},t}$ potentially results in a less biased sample from $p_\theta(\mathbf{z}|\mathbf{x}, t)$, whilst also providing more opportunities for the EBM to learn.

Additionally, a larger value of N_t was investigated. Aside from providing a better estimator for the log-marginal likelihood, this also results in a greater absolute number of partitions of the temperature schedule in regions where the prior is dominant in Eq. 17. The schedule is more finely divided, allowing for more evaluation points to be clustered around the lower temperatures, as shown in Fig. 2. Incorporating more intermediate distributions between the prior and the posterior can aid exploration of the loss landscape, which may help mitigate convergence to local optima. The $p = 0.1$ schedule is significantly more skewed towards the higher temperatures in Fig. 2, which corresponds to $p_\theta(\mathbf{z}|\mathbf{x}, t)$ being more representative of the true Bayesian posterior distribution than the prior distribution,

$p_\alpha(\mathbf{z})$. These upwards-skewed temperature schedules involve fewer instances of simpler distributions resembling the prior, limiting the model's ability to explore before confronting the more complicated posterior distribution. A larger N_t may mitigate this.

Furthermore, we also introduced a new parameter η to weight the $D_{\mathrm{KL}}(\cdot||\cdot)$ bias terms of Eq. 19:

$$\log p_\theta(\mathbf{x}) = \frac{1}{2}\sum_i \Delta t_i (E_{i-1} + E_i) + \eta \cdot \frac{1}{2}\sum_i D_{\mathrm{KL}}(p_{i-1}||p_i) + D_{\mathrm{KL}}(p_i||p_{i-1}) \tag{22}$$

This is no longer a true marginal likelihood evaluation, but it allows us to control the weighting applied to the $D_{\mathrm{KL}}(p_i||p_{i-1})$ terms, which corresponds to the error associated with the trapezium rule for numerical integration. Specifically, the KL divergences represent the discrepancies between adjacent tempered distributions in the temperature schedule (Eq. 17). Increasing η was hypothesised to allow the discrepancy between adjacent tempered distributions to exert a larger influence over the loss evaluation. These discrepancies may correspond to the appearance of local modes and added complexity within the intermediary distributions between the prior and the true Bayesian posterior distributions. A higher value of η may force the model to better navigate the increasing complexity as it transitions from the prior to the posterior, potentially improving performance if the default balance between exploration and exploitation is suboptimal. It may also further prioritise learning through the posterior MCMC loop, where the EBM is incorporated through the prior in Eq. 17.

5 Results

5.1 Thermodynamic Integration can sculpt the learning gradient

Figs. 3-4 showcase the different learning gradient variances achievable by the trained vanilla and altered models. The general relationship persisted for CIFAR-10. The boxplots demonstrate that different variances can be achieved by adjusting batch sizes, N_{batch}, for the vanilla model or by varying temperature powers, p, for the altered model.

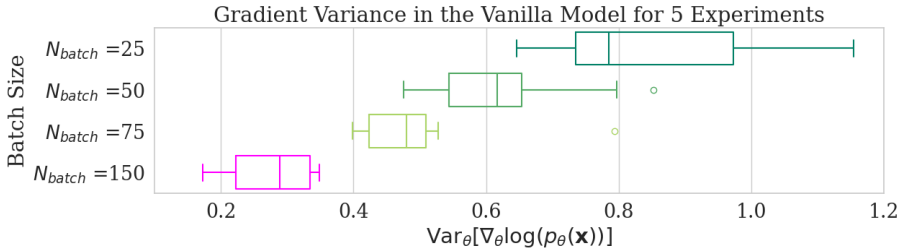


Figure 3: Relationship between final learning gradient variance of the vanilla model and batch size. The models were trained on CelebA for 50 epochs. Whilst there is a relationship between batch size and learning gradient variance, there is a large range across the 5 experiments, resulting in less predictable variance control.

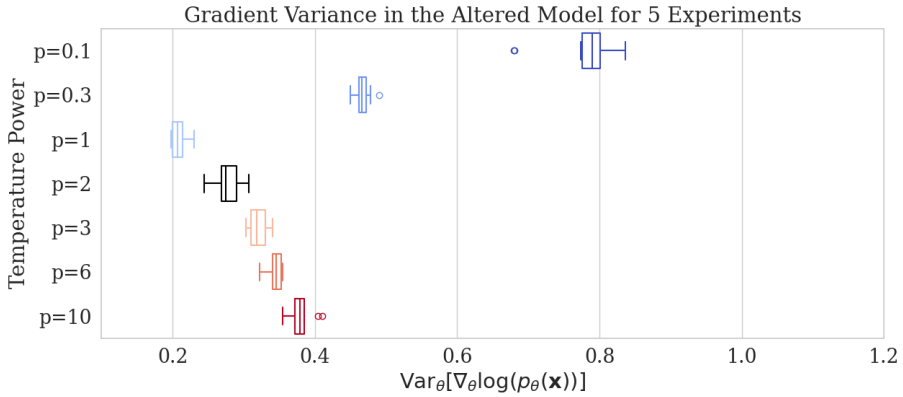


Figure 4: Relationship between final learning gradient variance and p in Eq. 21 for the altered model. The models were once again trained on CelebA for 50 epochs. The amount of discretisation (Eq. 18) is held fixed at $N_t = 10$. Larger variances are attainable with $0 < p \leq 1$. Fine-grained changes can be achieved with $p > 1$.

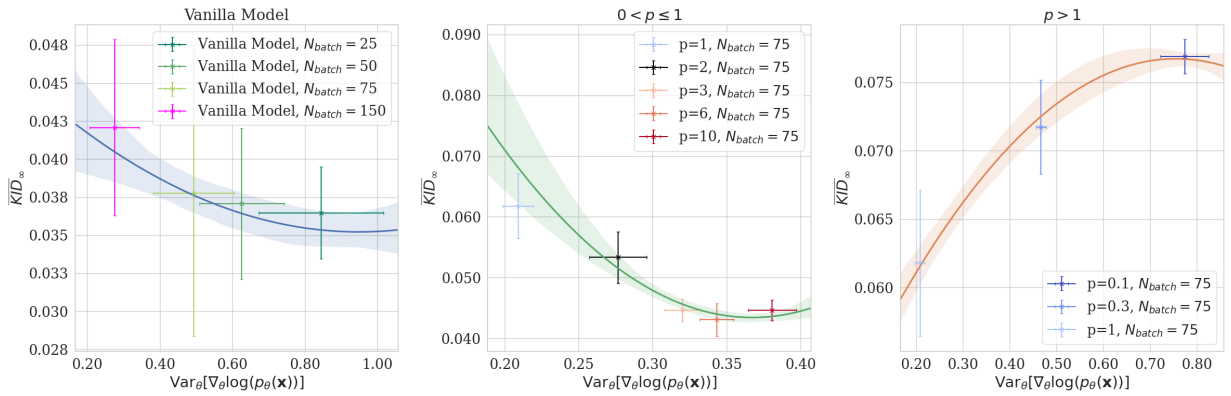


Figure 5: The relationships between learning gradient noise and generated image fidelity, as described by $\overline{KID}_\infty$. These results correspond to models trained on the CelebA dataset for 50 epochs. Quadratic regressions have been included to serve as fitting representations of the different regimes. Lower $\overline{KID}_\infty$ corresponds to higher quality generated samples. The means across repetitions are plotted, with error bars representing the standard deviations.

However, the range of variances observed across the five distinct repetitions is notably wider in Fig. 3 compared to Fig. 4. The model incorporating Thermodynamic Integration can therefore achieve learning gradient variances comparable to those achieved by adjusting batch sizes in the vanilla model, but with greater reliability across repetitions. Thermodynamic Integration offers a more robust and controllable means of achieving different learning gradient variances through carefully scheduling the temperatures.

5.2 There is a striking relationship between image fidelity and learning gradient variance

Shown in Fig. 5 are the results for the different vanilla and altered models incorporating the outlined experimental setups. In the vanilla model, batch size was used to tune learning gradient variance. In the altered model incorporating Thermodynamic Integration, the batch size was held at 75, whilst p in Eq. 21 was used to alter learning gradient variance.

In terms of image fidelity, the vanilla model outperformed the altered model, despite the large computational cost incurred by the adoption of Thermodynamic Integration. A higher discretisation, (greater than $N_t = 10$), typically improves image fidelity, however it also entails an even greater computational expense. This implies that Thermodynamic Integration ought to be employed primarily when exact management of learning gradient variance is required, rather than for the sake of image quality.

In the $p > 1$ regime, the altered model attains comparable learning gradient variances as the vanilla model with $N_{\text{batch}} = 150$. Further increasing p beyond 1 increases the learning gradient variance, and seemingly enhances generative capacity, (resulting in lower $\overline{KID}_\infty$), until a minimum is achieved at $p = 6$.

Quadratic regressions have been applied to the results. They demonstrate that there is a striking dependence between image fidelity and gradient variance. However, the relationship must first be divided into different regimes depending on the form of temperature schedule used.

They also show that generative capacity is generally improved with larger learning gradient variances for the vanilla models. Although cross-referencing against the results for CIFAR-10 suggests that this trend holds only up to a certain variance, beyond which increasing gradient noise deteriorates image quality. Similarly for the altered models with $p > 1$, a particular variance is required to achieve a clear maximum image fidelity, (around 0.35).

Importantly, a large discrepancy arises between the models in Fig. 5 when the altered model operates within the $0 < p \leq 1$ regime. For comparable variances in Fig. 5, the altered model yields lower-quality images than the vanilla model. This suggests that learning gradient variance is not the sole influence regarding the model's generative capacity.

A standout example is $p = 0.1$, which exhibits a similar learning gradient variance as the vanilla model with $N_{batch} = 25$, but demonstrates comparably worse images and a larger $\overline{KID}_\infty$, which stands as motivation for investigating other Thermodynamic Integration parameters.

Cross-referencing against training loss curves suggests that this discrepancy is attributed to mode collapse. The altered models with $0 < p \leq 1$ converged to a local optima rather than the true minimum in the loss landscape.

5.3 Tuning parameters can slightly mitigate an imbalance between exploration and exploitation.

Fig. 6 presents the models with the alterations discussed in Section 4 overlaid on the previous results for reference. The generated image samples suggest that all $p = 0.1$ models still suffered from mode collapse, which indicates that the discrepancy is a result of the form of the $p = 0.1$ schedule in Fig. 2 rather than any tunable parameters.

Firstly, increasing $K_{\mathbf{z}|\mathbf{x},t}$ from 20 to 40 resulted in similar, if not worse, image quality. This suggests that the EBM's learning capacity is not limited by the MCMC sampling procedure. The mode collapse evident for the $p = 0.1$ model is therefore unlikely to be attributed to poor exploration or bias of the unadjusted Langevin algorithm.

Increasing the schedule discretisation parameter, N_t, and the weighting applied to the $D_{\mathrm{KL}}(\cdot||\cdot)$ terms, η, improved image fidelity. The improvement in image fidelity for increased N_t, despite the reduction in learning gradient variance, reinforces the claim that learning gradient variance is not the sole contributor to image fidelity.

This suggests that the generative capacity of the altered model is directly dependent on the balance between exploration and exploitation provided by the temperature schedule. A larger N_t allows the model to step through more intermediary distributions between the simpler prior and more complex posterior distributions, enhancing its ability to explore the loss landscape. Increasing the weighting of the discrepancy terms in Eq. 22 with η may have emphasised the added complexity or local modes between adjacent tempered distributions, potentially causing the model to navigate them more aggressively.

Increasing η emerges as the most promising approach towards enhancing image fidelity in the model incorporating Thermodynamic Integration at no additional cost. However, it leads to an inexact marginal likelihood evaluation. On the other hand, increasing N_t yields a similar positive effect whilst incurring significant additional computational overhead. However, both changes were unable to prevent mode collapse, evident through the generated images.

This issue might be attributed to the small increases applied to N_t and η. Perhaps a much larger discretisation of the temperature schedule is needed to allow the model to escape the local minima entirely. However, this requires an infeasible amount of training time. Alternatively, it could be that that N_t and η are more appropriate for fine-tuning image fidelity once a minimum in the loss landscape has been converged to. The underlying issue of mode collapse might only be solvable by a temperature schedule that favors partitions clustered towards the simpler prior.

Overall, the results suggest that compared to the temperature schedule governed by p, varying $K_{\mathbf{z}|\mathbf{x},t}$ produced a statistically insignificant impact. In contrast, the temperature discretisation N_t and KL divergence weighting η had a more significant impact on improving and fine-tuning image quality, but did not help the model escape local optima. Larger values for N_t are likely to further enhance the model's explorative capacity and perhaps even help it escape local modes. However, this comes at a high cost. The 50-epochs training time is increased by a factor of 3 from the baseline $p = 0.1$ model with the increase from $N_t = 10$ to $N_t = 30$.

6 Conclusions

This study demonstrates the importance of adopting a distributional perspective on the learning gradient to enhance the capabilities of deep generative models and the quality of the images they produce. It presents Thermodynamic Integration as a powerful method for controlling learning gradient variance and shaping the balance between exploration and exploitation. However, it also shows that while Thermodynamic Integration provides these advantages, it comes with a significant computational cost and does not achieve better image fidelity compared to traditional methods of estimating the marginal likelihood, especially when limited schedule discretisation is used.

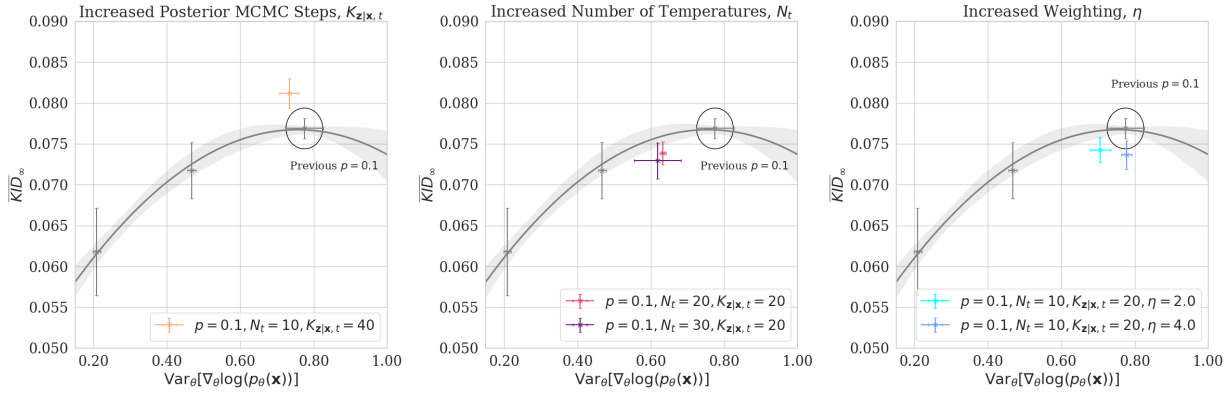


Figure 6: The models with the alterations discussed in Section 4 are overlaid on the previous data in the $0 < p \leq 1$ regime for comparison. The model with $p = 0.1, N_t = 10, K_{\mathbf{z}|\mathbf{x},t} = 20$ has been circled to serve as a comparative baseline. The means are plotted, with error bars representing the standard deviations across the 5 repetitions.

References

[1] Mikołaj Bińkowski et al. *Demystifying MMD GANs*. 2021. arXiv: 1801.01401 [stat.ML].

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. *Large Scale GAN Training for High Fidelity Natural Image Synthesis*. 2019. arXiv: 1809.11096 [cs.LG].

[3] Ben Calderhead and Mark Girolami. "Estimating Bayes factors via thermodynamic integration and population MCMC". In: *Computational Statistics & Data Analysis* 53.12 (2009), pp. 4028–4045. ISSN: 0167-9473. DOI: https://doi.org/10.1016/j.csda.2009.07.025. URL: https://www.sciencedirect.com/science/article/pii/S0167947309002722.

[4] Min Jin Chong and David Forsyth. *Effectively Unbiased FID and Inception Score and where to find them*. 2020. arXiv: 1911.07023 [cs.CV].

[5] Fartash Faghri et al. *A Study of Gradient Variance in Deep Learning*. 2020. arXiv: 2007.04532 [cs.LG].

[6] N. Friel and A. N. Pettitt. "Marginal Likelihood Estimation via Power Posteriors". In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 70.3 (2008), pp. 589–607. ISSN: 13697412, 14679868. URL: http://www.jstor.org/stable/20203843 (visited on 10/26/2023).

[7] Martin Heusel et al. *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium*. 2018. arXiv: 1706.08500 [cs.LG].

[8] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[9] Elizabeth Koehler, Elizabeth Brown, and Sebastien Haneuse. "On the Assessment of Monte Carlo Error in Simulation-Based Statistical Analyses". In: *The American statistician* 63 (May 2009), pp. 155–162. DOI: 10.1198/tast.2009.0030.

[10] Matthias Wright and Hayden Donnelly and Boris Dayma and Saurav Maheshkar. *jax-fid: FID computation in Jax/Flax*. https://github.com/matthias-wright/jax-fid. License: Apache-2.0 License. Year of Last Update.

[11] Arvind Neelakantan et al. *Adding Gradient Noise Improves Learning for Very Deep Networks*. 2015. arXiv: 1511.06807 [stat.ML].

[12] Bo Pang et al. "Learning Latent Space Energy-Based Prior Model". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 21994–22008. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/fa3060edb66e6ff4507886f9912e1ab9-Paper.pdf.

[13] Herbert E. Robbins. "A Stochastic Approximation Method". In: *Annals of Mathematical Statistics* 22 (1951), pp. 400–407. URL: https://api.semanticscholar.org/CorpusID:16945044.

[14] Christian Szegedy et al. *Rethinking the Inception Architecture for Computer Vision*. 2015. arXiv: 1512.00567 [cs.CV].

7 Appendix

7.1 Codebase

The study was implemented in JAX at `https://github.com/PritRaj1/JAX-ThermoEBM`. The raw experimental readings have also been provided as part of the codebase.

7.2 Summary of findings

7.2.1 Learning gradient variance and image fidelity

Param.	Learning Gradient Noise	Image Fidelity	Controllable?
N_{batch}	Decreasing N_{batch} increases $\text{Var}_\theta\left[\nabla_\theta \mathcal{L}(\theta, \mathbf{x})\right]$	Decreasing N_{batch} improves quality until a minimum is achieved, (depending on the dataset being learned).	No - large range across repetitions
$p > 1$	Increasing p increases $\text{Var}_\theta\left[\nabla_\theta \mathcal{L}(\theta, \mathbf{x})\right]$	Increasing p improves quality until a minimum is achieved at $p \approx 6$. Increasing further reduces quality.	Yes - small range across repetitions
$0 < p \leq 1$	Decreasing p increases $\text{Var}_\theta\left[\nabla_\theta \mathcal{L}(\theta, \mathbf{x})\right]$	Much poorer image quality due to mode collapse. Increasing p improves quality	Yes - small range across repetitions

Table 1: The relationships between learning gradient variance, image fidelity, batch size and temperature schedule.

7.2.2 Importance of Thermodynamic Integration parameters

Param.	Learning Gradient Noise	Image Fidelity	Significance
p	(See Tab. 1)	(See Tab. 1)	Upmost significance - defines the critical balance between exploration and exploitation
N_t	Increasing N_t decreases $\text{Var}_\theta\left[\nabla_\theta \mathcal{L}(\theta, \mathbf{x})\right]$	Increasing N_t improves quality	Fair significance - improves model performance, but at a large cost
$K_{\mathbf{z}\|\mathbf{x},t}$	Increasing $K_{\mathbf{z}\|\mathbf{x},t}$ decreases $\text{Var}_\theta\left[\nabla_\theta \mathcal{L}(\theta, \mathbf{x})\right]$	No impact or increasing $K_{\mathbf{z}\|\mathbf{x},t}$ may even worsen image quality	More experimental work is required, but the results suggest that short run MCMC is suitable enough. Increasing $K_{\mathbf{z}\|\mathbf{x},t}$ comes at a computational cost
η	Increasing η decreases $\text{Var}_\theta\left[\nabla_\theta \mathcal{L}(\theta, \mathbf{x})\right]$	Increasing η slightly improves quality at no computational cost	More experimental work is required, but the results suggest that η may be a promising means of forcing the model to navigate adjacent tempered distributions more carefully

Table 2: Summary of the influence of different Thermodynamic Integration parameters.

7.3 Experiment setup

7.3.1 Hyperparameters

Param.	Codebase Var. Name	Value
Dataset	DATASET	CelebA
Number of training epochs	NUM_EPOCHS	50
Batch size	BATCH_SIZE	75 (variable for vanilla model)
Number of training examples	NUM_TRAIN_DATA	12000
Number of testing examples	NUM_VAL_DATA	4500
Number of repetitions to conduct	NUM_EXPERIMENTS	5
Latent space dimension	Z_CHANNELS	80
Complexity of EBM, (nez in tab. 5)	EBM_FEATURE_DIM	100
EBM Leaky-ReLU leak coefficient	EBM_LEAK	0.1
Complexity of GEN, (ngz in tab. 5)	GEN_FEATURE_DIM	64
GEN Leaky-ReLU leak coefficient	GEN_LEAK	0.2
σ_0 (for $\pi_0(\mathbf{z})$ in Eq. 1)	p0_SIGMA	1
σ_l (for ϵ in eq 2)	LKHOOD_SIGMA	0.3
Initial learning rate for α	E_INITIAL_LR	0.00002
Initial learning rate for β	G_INITIAL_LR	0.0001
Final learning rate for α	E_FINAL_LR	0.00001
Final learning rate for β	G_FINAL_LR	0.00005
Optimiser for both networks	(-)	Adam
EBM adam opt. beta 1	E_BETA_1	0.5
EBM adam opt. beta 2	E_BETA_2	0.999
GEN adam opt. beta 1	G_BETA_1	0.5
GEN adam opt. beta 2	G_BETA_2	0.999
LR scheduler	(-)	Exponential decay
LR scheduler start	BEGIN_EPOCH	1
LR scheduler decay	DECAY_RATE	0.975
LR schedule step	STEP_INTERVAL	1
MCMC sampler	(-)	Unadjusted Langevin
s for $p_\alpha(\mathbf{z})$ (in Eq. 10)	E_STEP_SIZE	0.16
s for $p_\theta(\mathbf{z}\|\mathbf{x}, t)$ (in Eq. 10)	G_STEP_SIZE	0.01
$K_\mathbf{z}$ for $p_\alpha(\mathbf{z})$ (in Eq. 10)	E_SAMPLE_STEPS	60
$K_{\mathbf{z}\|\mathbf{x},t}$ for $p_\theta(\mathbf{z}\|\mathbf{x}, t)$ (in Eq. 10)	G_SAMPLE_STEPS	20
p (in Eq. 21)	TEMP_POWER	0 for vanilla model, variable otherwise
N_t (in Eq. 18)	NUM_TEMPS	10
$D_{\mathrm{KL}}(\cdot\|\|\cdot)$ weight, η in Eq. 22	KL_BIAS_WEIGHT	1

Table 3: Hyperparamter setup for CelebA

Param.	Codebase Var. Name	Value
Dataset	DATASET	CIFAR10
Number of training epochs	NUM_EPOCHS	50
Batch size	BATCH_SIZE	75 (variable for vanilla model)
Number of training examples	NUM_TRAIN_DATA	12000
Number of testing examples	NUM_VAL_DATA	4500
Number of repetitions to conduct	NUM_EXPERIMENTS	5
Latent space dimension	Z_CHANNELS	100
Complexity of EBM, (nez in tab. 5)	EBM_FEATURE_DIM	100
EBM Leaky-ReLU leak coefficient	EBM_LEAK	0.1
Complexity of GEN, (ngz in tab. 5)	GEN_FEATURE_DIM	64
GEN Leaky-ReLU leak coefficient	GEN_LEAK	0.2
σ_0 (for $\pi_0(\mathbf{z})$ in Eq. 1)	p0_SIGMA	1
σ_l (for ϵ in eq 2)	LKHOOD_SIGMA	0.3
Initial learning rate for α	E_INITIAL_LR	0.00002
Initial learning rate for β	G_INITIAL_LR	0.0001
Final learning rate for α	E_FINAL_LR	0.00001
Final learning rate for β	G_FINAL_LR	0.00005
Optimiser for both networks	(-)	Adam
EBM adam opt. beta 1	E_BETA_1	0.5
EBM adam opt. beta 2	E_BETA_2	0.999
GEN adam opt. beta 1	G_BETA_1	0.5
GEN adam opt. beta 2	G_BETA_2	0.999
LR scheduler	(-)	Exponential decay
LR scheduler start	BEGIN_EPOCH	1
LR scheduler decay	DECAY_RATE	0.975
LR schedule step	STEP_INTERVAL	1
MCMC sampler	(-)	Unadjusted Langevin
s for $p_\alpha(\mathbf{z})$ (in Eq. 10)	E_STEP_SIZE	0.16
s for $p_\theta(\mathbf{z}\|\mathbf{x}, t)$ (in Eq. 10)	G_STEP_SIZE	0.01
$K_\mathbf{z}$ for $p_\alpha(\mathbf{z})$ (in Eq. 10)	E_SAMPLE_STEPS	60
$K_{\mathbf{z}\|\mathbf{x},t}$ for $p_\theta(\mathbf{z}\|\mathbf{x}, t)$ (in Eq. 10)	G_SAMPLE_STEPS	40
p (in Eq. 21)	TEMP_POWER	0 for vanilla model, variable otherwise
N_t (in Eq. 18)	NUM_TEMPS	10
$D_{\mathrm{KL}}(\cdot\|\|\cdot)$ weight, η in Eq. 22	KL_BIAS_WEIGHT	1

Table 4: Hyperparamter setup for CIFAR-10

7.3.2 Network architectures

Model	Layers	In-Out Size, Stride
	Input: z	Z_CHANNELS
EBM	Linear, LReLU	nez -
for both	Linear, LReLU	nez -
	Linear	1 -
	Input: z	1x1xZ_CHANNELS
	4x4 convT(ngf x 16), LReLU	4x4x(ngf x 16), 1
GEN for	4x4 convT(ngf x 8), LReLU	4x4x(ngf x 8), 2
CIFAR-10	4x4 convT(ngf x 4), LReLU	4x4x(ngf x 4), 2
	4x4 convT(ngf x 2), LReLU	4x4x(ngf x 2), 2
	4x4 convT(3), Tanh	4x4x3, 2
	Input: z	1x1xZ_CHANNELS
	4x4 convT(ngf x 16), LReLU	4x4x(ngf x 16), 1
	4x4 convT(ngf x 8), LReLU	4x4x(ngf x 8), 2
GEN for CelebA	4x4 convT(ngf x 4), LReLU	4x4x(ngf x 4), 2
	4x4 convT(ngf x 2), LReLU	4x4x(ngf x 2), 2
	4x4 convT(ngf x 1), LReLU	4x4x(ngf x 1), 2
	4x4 convT(3), Tanh	4x4x3, 2

Table 5: Model architectures for both datasets. These have been simplified compared to those in [12]. Unlike the U-Net architecture's decoder used for the generator in [12], our generator models maintain a consistent kernel size throughout. These adjustments do not impact the results and conclusions, which remain comparative across different models.

7.4 Derivation of the thermodynamic integral

$$p_\theta(\mathbf{z}|\mathbf{x},t) = \frac{p_\beta(\mathbf{x}|\mathbf{z})^t p_\alpha(\mathbf{z})}{\mathcal{Z}(\mathbf{x}|t)}, \quad \text{where} \quad \mathcal{Z}(\mathbf{x}|t) = \int p_\beta(\mathbf{x}|\mathbf{z})^t p_\alpha(\mathbf{z})\, d\mathbf{z}$$

$$\begin{aligned}
\frac{d}{dt}\log(\mathcal{Z}(\mathbf{x}|t)) &= \frac{1}{\mathcal{Z}(\mathbf{x}|t)}\frac{d}{dt}\mathcal{Z}(\mathbf{x}|t) \\
&= \frac{1}{\mathcal{Z}(\mathbf{x}|t)}\int \frac{d}{dt}\left(p_\beta(\mathbf{x}|\mathbf{z})^t\right) p_\alpha(\mathbf{z})\, d\mathbf{z} \\
&= \frac{1}{\mathcal{Z}(\mathbf{x}|t)}\int p_\beta(\mathbf{x}|\mathbf{z})^t \log(p_\beta(\mathbf{x}|\mathbf{z})) p_\alpha(\mathbf{z})\, d\mathbf{z} \\
&= \int \frac{p_\beta(\mathbf{x}|\mathbf{z})^t \log(p_\beta(\mathbf{x}|\mathbf{z})) p_\alpha(\mathbf{z})}{\mathcal{Z}(\mathbf{x}|t)}\, d\mathbf{z}
\end{aligned}$$

Given $\mathcal{Z}(\mathbf{x}|t=0)$ is the prior marginalised over $\mathbf{z}$, which is equal to 1, and $\mathcal{Z}(\mathbf{x}|t=1)$ is the marginal likelihood:

$$\begin{aligned}
\frac{1}{\mathcal{Z}(\mathbf{x}|t)}\frac{d}{dt}\mathcal{Z}(\mathbf{x}|t) &= \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}|\mathbf{z})\right] \\
\frac{d}{dt}\log(\mathcal{Z}(\mathbf{x}|t)) &= \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}|\mathbf{z})\right] \\
\int_0^1 \frac{d}{dt}\log(\mathcal{Z}(\mathbf{x}|t))\,dt &= \int_0^1 \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}|\mathbf{z})\right]\,dt \\
\log(\mathcal{Z}(\mathbf{x}|t))\Big|_{t=0}^{t=1} &= \int_0^1 \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}|\mathbf{z})\right]\,dt \\
\log(\mathcal{Z}(\mathbf{x}|t=1)) - \log(\mathcal{Z}(\mathbf{x}|t=0)) &= \int_0^1 \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}|\mathbf{z})\right]\,dt \\
\log p_\theta(\mathbf{x}) - \log(1) &= \int_0^1 \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}|\mathbf{z})\right]\,dt \\
\log p_\theta(\mathbf{x}) &= \int_0^1 \mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}|\mathbf{z})\right]\,dt
\end{aligned}$$

7.5 Summary of differences between the vanilla and altered models

Term	Vanilla Model	Altered Model
$\log(p_\theta(\mathbf{x}))$	$\mathbb{E}_{p_\theta(\mathbf{z}\|\mathbf{x})}\left[\log(p_\alpha(\mathbf{z})) + \log(p_\beta(\mathbf{x}\|\mathbf{z}))\right]$	$\int_0^1 \mathbb{E}_{p_\theta(\mathbf{z}\|\mathbf{x},t)}\left[\log p_\beta(\mathbf{x}\|\mathbf{z})\right]\,dt$
$p_\theta(\mathbf{z}\|\mathbf{x},t)$	$\frac{p_\beta(\mathbf{x}\|\mathbf{z})p_\alpha(\mathbf{z})}{\mathcal{Z}(\mathbf{x})}$	$\frac{p_\beta(\mathbf{x}\|\mathbf{z})^t p_\alpha(\mathbf{z})}{\mathcal{Z}(\mathbf{x}\|t)}$
$\nabla_\mathbf{z}\log p_\theta(\mathbf{z}\|\mathbf{x},t)$	$-\nabla_\mathbf{z}\frac{\|\mathbf{x}-g_\beta(\mathbf{z})\|^2}{2\sigma_l^2} + \nabla_\mathbf{z} f_\alpha(\mathbf{z}) - \frac{\mathbf{z}}{\sigma_0^2}$	$-\nabla_\mathbf{z}\frac{t\cdot\|\mathbf{x}-g_\beta(\mathbf{z})\|^2}{2\sigma_l^2} + \nabla_\mathbf{z} f_\alpha(\mathbf{z}) - \frac{\mathbf{z}}{\sigma_0^2}$
$\nabla_\alpha\mathcal{L}(\theta,\mathbf{x})$	Tracked through Langevin sampling for $p_\theta(\mathbf{z}\|\mathbf{x})$ and explicit dependence on $\log(p_\alpha(\mathbf{z}))$, which results in a prior matching formulation in Eq. 7.	Solely tracked through Langevin sampling for $p_\theta(\mathbf{z}\|\mathbf{x},t)$.

Table 6: Key differences between the vanilla model and the altered model incorporating Thermodynamic Integration.

7.6 Further details regarding image fidelity metrics

Quantifying the perceptual quality of an image dataset poses a significant challenge, leading to the emergence of multiple metrics across machine learning literature. However, none of these metrics are flawless. They each come with their own equations, such as those depicted in 23 and 24, and introduce biases that are contingent on the number of samples used. For instance, computing the mean in Eq. 23 necessitates a Monte Carlo estimate, which carries an associated error as shown in Eq. 13.

Among the metrics widely embraced by the community are Fréchet Inception Distance (FID) [7] and Kernel Inception Distance (KID) [1]. These metrics, detailed underneath, are commonly employed despite their limitations.

To extract feature representations from images, InceptionV3 was used, a convolutional neural network (CNN) pre-trained on the ImageNet dataset [14]. For the JAX implementation of this study, the pre-trained InceptionV3 implementation was taken from the work of [10]. When an image is inputted into InceptionV3, it undergoes multiple layers of convolutional and pooling operations, thereby capturing features at various levels of abstraction. These extracted features are subsequently used in the equations provided below to compute the metrics.

7.6.1 FID

$$\text{FID} = \|\mu_{\text{real}} - \mu_{\text{gen}}\|^2 + \text{Tr}(\Sigma_{\text{real}} + \Sigma_{\text{gen}} - 2\sqrt{\Sigma_{\text{real}}\Sigma_{\text{gen}}}) \tag{23}$$

Here, μ_{real} and μ_{gen} represent the mean feature representations of real and generated samples, respectively, obtained via InceptionV3. Similarly, Σ_{real} and Σ_{gen} denote their respective covariance matrices.

7.6.2 KID

$$\text{KID} = \frac{1}{n^2}\text{Tr}(H\tilde{H}) - \frac{2}{n}\text{Tr}(HK) + \text{Tr}(KK) \tag{24}$$

where n is the number of samples, H is the Gram matrix computed from the feature representations of real samples, $\tilde{H}$ is the Gram matrix computed from the feature representations of generated samples, and K is the cross-gram matrix between real and generated samples.

Here, the Gram matrix $\tilde{H}$ was computed using the radial basis function (RBF) kernel. Each entry $\tilde{H}_{ij}$ of the matrix is computed as the RBF kernel function applied to the Euclidean distance between the feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$. Specifically, for $\tilde{H}$, the RBF kernel $K(\mathbf{x}_i, \mathbf{x}_j)$ with length scale $\gamma = \frac{1}{\text{num\_features}}$ was used:

$$\tilde{H}_{ij} = \exp\left(-\frac{\|\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\|^2}{2\gamma^2}\right)$$

Here, $\|\cdot\|^2$ represents the squared Euclidean distance between the feature vectors $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{x}}_j$ of the generated images, obtained from the InceptionV3 model.

7.6.3 Unbiased image metrics for a reliable experimental method

To gauge the perceptual quality of images generated by the latent space energy-based model in a manner resilient to the limitations outlined earlier, both the FID and the KID metrics are tracked. This dual approach allows for a more comprehensive assessment of image quality while acknowledging and mitigating the biases inherent in each metric. However, it was ultimately discovered that KID was the more reliable metric, as detailed later in this section. Therefore, FID was disregarded in the interpretation of results within Section 5.

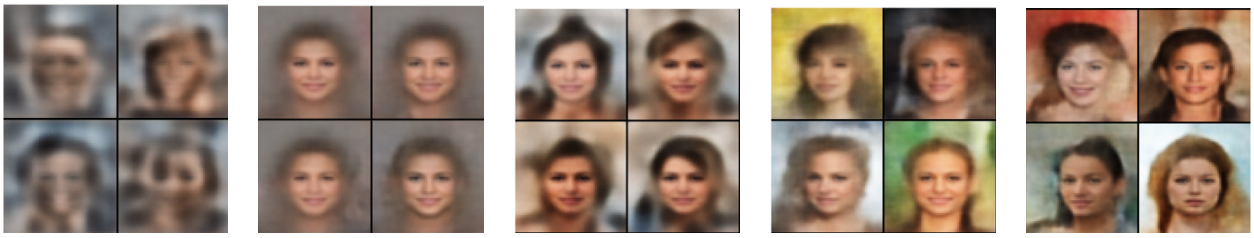
Furthermore, to address the bias in these metrics as mentioned earlier, a scheme inspired by the prior work of [4] is adopted. In this approach, FID readings are gathered for various sample sizes, and then linear regression is used to extrapolate to an infinitely-sized sample set. This technique yields effectively unbiased estimators of the metrics, enhancing the robustness of the perceptual quality evaluation process.

In our experiment, to implement this scheme for both FID and KID, the following sample set sizes are employed:

$$\begin{aligned}\text{Samples Sizes} :\{&1000, 1389, 1778, 2167, 2556,\\ &2944, 3333, 3722, 4111, 4500\}\end{aligned}$$

Linear regression is subsequently applied to evaluate the unbiased estimators, $\overline{FID}_\infty$ and $\overline{KID}_\infty$. In general, $\overline{KID}_\infty$ emerged as the more robust metric across all datasets. Figs. 7 and 8 present visual examples to illustrate this observation.

Specifically, throughout the study, generative performance and their relation to the metrics was verified through human inspection of recorded image samples. In Figs. 7 and 8, each grid of images corresponds to different models at various stages of their training runs, each chosen to exemplify how the metrics assess the perceptual quality of different images. For both datasets, the models used to generate samples were of substantially reduced complexity compared to the original implementation in [12].



(a) $\overline{FID}_\infty = 380.5$, $\overline{KID}_\infty = 0.102$ (b) $\overline{FID}_\infty = 246.3$, $\overline{KID}_\infty = 0.075$ (c) $\overline{FID}_\infty = 268.1$, $\overline{KID}_\infty = 0.067$ (d) $\overline{FID}_\infty = 275.0$, $\overline{KID}_\infty = 0.050$ (e) $\overline{FID}_\infty = 252.5$, $\overline{KID}_\infty = 0.033$

Figure 7: Performance of unbiased metrics against samples of the CelebA dataset, generated by various implementations of the vanilla and altered models at different stages of their 50-epoch training runs. This collection of chosen images serves to illustrate why $\overline{KID}_\infty$ is considered to be more reliable. For instance, in Fig. 7b, despite exhibiting the lowest $\overline{FID}_\infty$, suggesting superior image quality, the grid of images is subjectively less impressive than Figs. 7c, 7d, and 7e. In contrast, $\overline{KID}_\infty$ was more representative of what can be considered subjectively better quality.

(a) $\overline{FID}_\infty = 492.7$, $\overline{KID}_\infty = 0.084$ (b) $\overline{FID}_\infty = 456.2$, $\overline{KID}_\infty = 0.074$ (c) $\overline{FID}_\infty = 464$, $\overline{KID}_\infty = 0.052$ (d) $\overline{FID}_\infty = 454.2$, $\overline{KID}_\infty = 0.046$ (e) $\overline{FID}_\infty = 459$, $\overline{KID}_\infty = 0.039$

Figure 8: Performance of unbiased metrics against samples of the CIFAR-10 dataset, generated by our different implementations of the vanilla and altered models at different stages of their 50-epoch training runs.. Generally, with reduced complexity, the model struggled with the diversity of CIFAR-10. However, once again, the provided examples subjectively demonstrate why $\overline{KID}_\infty$ is considered more reliable for this study. For example, Fig. 8b is attributed a fairly low $\overline{FID}_\infty$, despite being comparatively worse in quality than Figs. 8c, 8d, and 8e.

7.7 Supplementary figures

7.7.1 Train loss

The loss curves in Fig. 9 exhibit notable variability due to normalisation, which has been applied to enable a easier comparison between the vanilla and altered models.

The normalisation removes an inherent bias that accumulates throughout the thermodynamic integral, and offsets the models with unnormalised log-probability distributions from each other.

These offsets make it difficult to compare the models without normalisation. However, as shown, the normalisation results in variability in the loss curves, attributed to the discrete means used in the normalisation process, introducing associated errors similar to the discretised estimator of the mean error described in Eq. 14.

The most important illustration from the figure is that the models were generally able to converge within 50 epochs, and that $p = 0.1$ and $p = 0.3$ have seemingly converged to a local optima, suggesting mode-collapse.

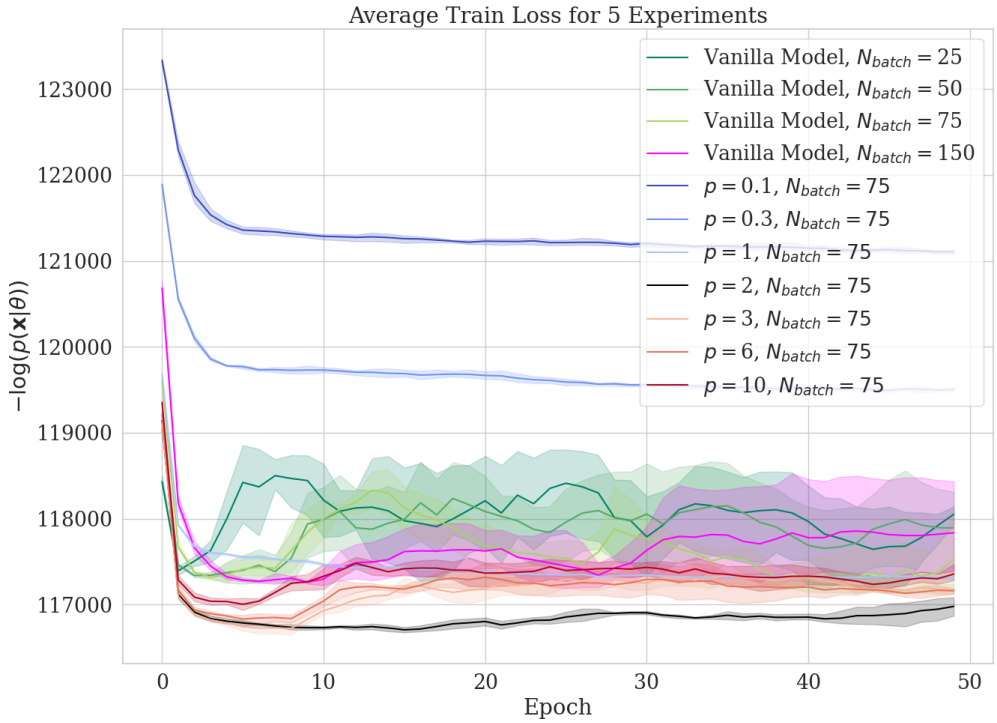


Figure 9: Evolution of training loss across five repetitions for each model trained on the CelebA dataset for 50 epochs.

7.7.2 Collated plot for clearer comparison

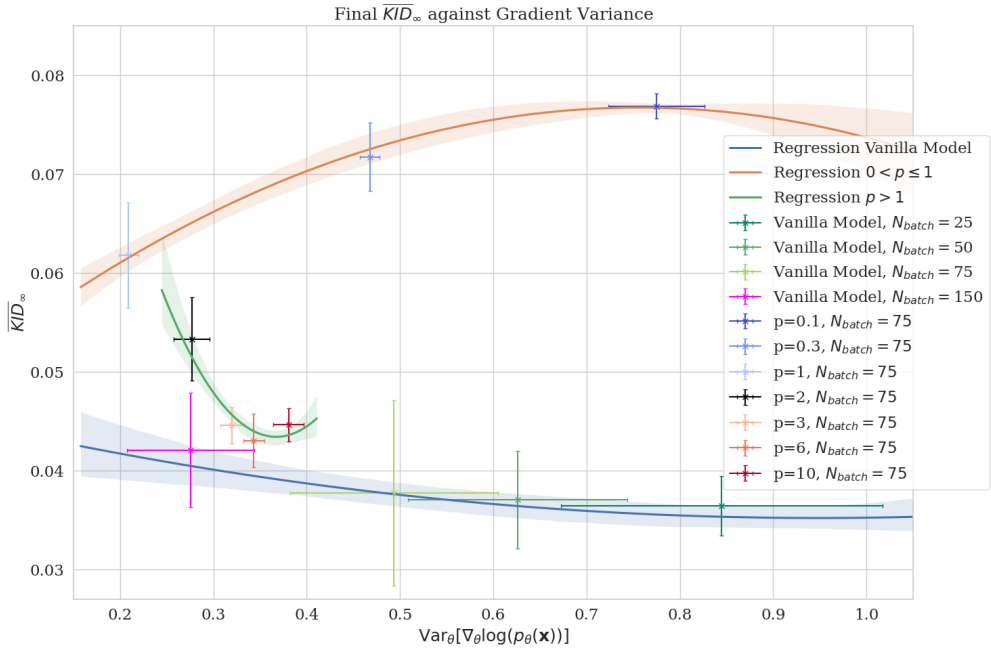
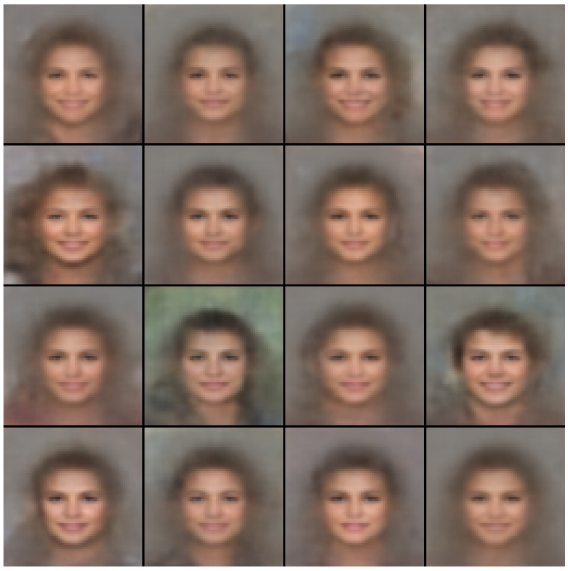


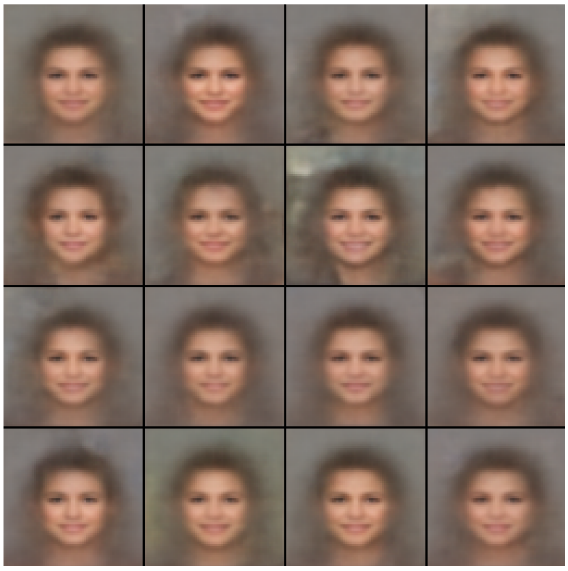
Figure 10: The relationships between learning gradient noise and generated image fidelity, as quantified by $\overline{KID}_\infty$ for CelebA. This plot, equivalent to Fig. 5, has been collated into one plot for comparison between the different regimes.

7.7.3 Collection of images demonstrating mode collapse

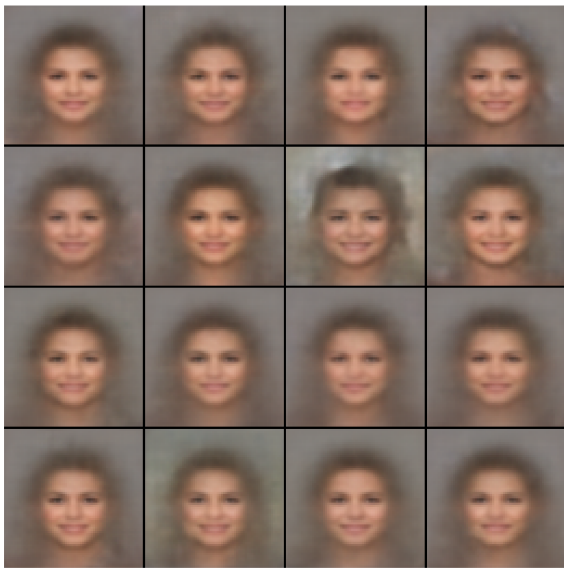
(a)
$p = 6, N_t = 10, K_{\mathbf{z}|\mathbf{x},t} = 20$



(b)
$p = 0.1, N_t = 30, K_{\mathbf{z}|\mathbf{x},t} = 20$



(c)
$p = 0.1, N_t = 10, K_{\mathbf{z}|\mathbf{x},t} = 40$



(d)
$p = 0.1, N_t = 10, K_{\mathbf{z}|\mathbf{x},t} = 20, \eta = 2.0$

Figure 11: CelebA images generated by the models of Fig. 6. All $p = 0.1$ models have suffered from mode collapse, in comparison to the $p = 6$ model, which has been included as reference. Changing the Thermodynamic Integration parameters does not help the altered model escape the local optima it has converged to.

7.7.4 $\overline{FID}_\infty$ results

As discussed in the main body, $\overline{FID}_\infty$ was found to be a less reliable metric than $\overline{KID}_\infty$, when visually inspecting the generated images. However, the results incorporating $\overline{FID}_\infty$ have been included here for completeness. These are generally more sporadic and erroneous than the well-defined results presented in Section 4.

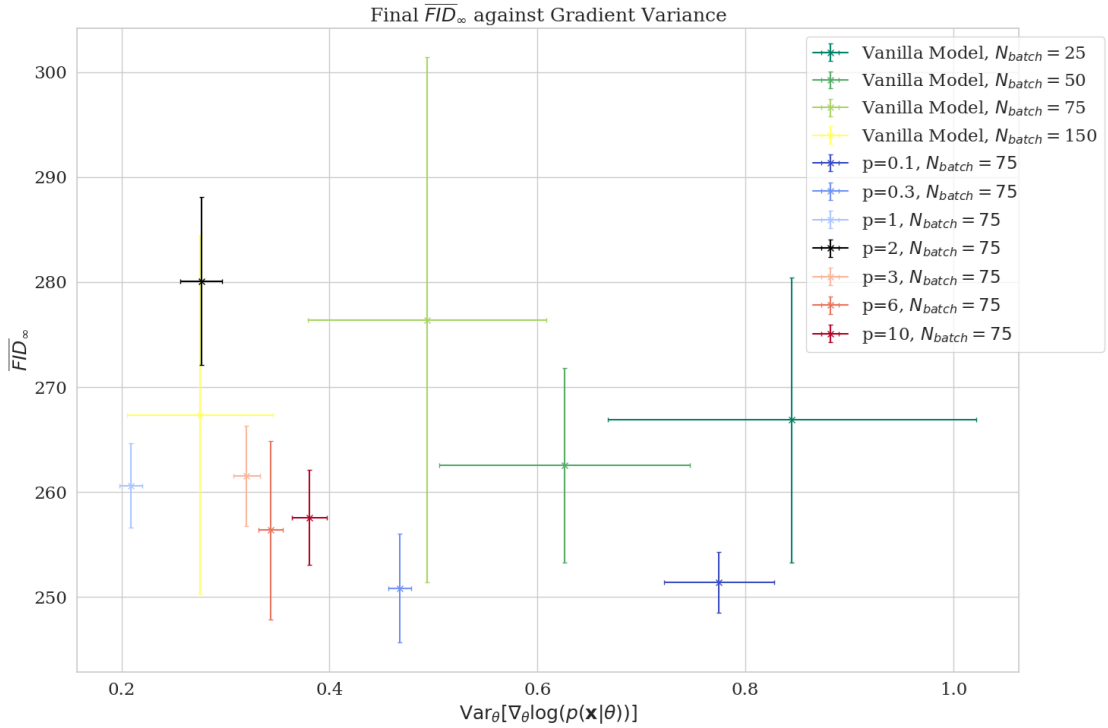


Figure 12: The effect of learning gradient variance on $\overline{FID}_\infty$ for the models trained on CelebA.

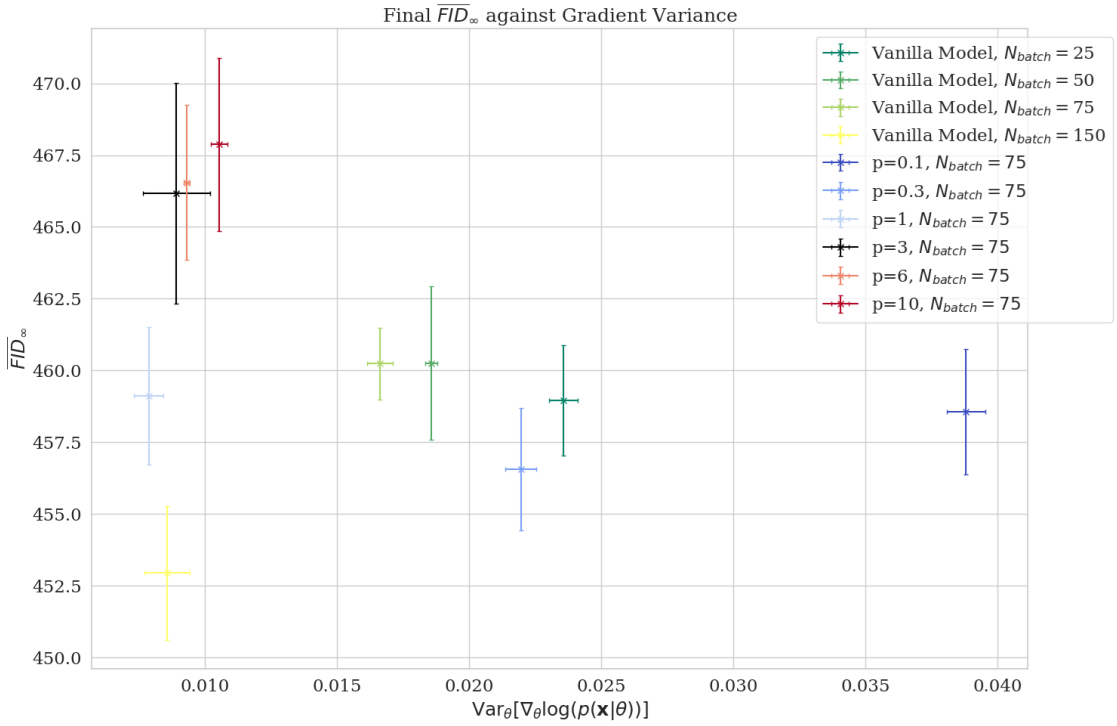


Figure 13: The effect of learning gradient variance on $\overline{FID}_\infty$ for the models trained on CIFAR-10.

7.7.5 Computational cost

Presented below are the associated computational costs for the models trained on CelebA. Thermodynamic Integration, whilst powerful, requires substantially more compute time.

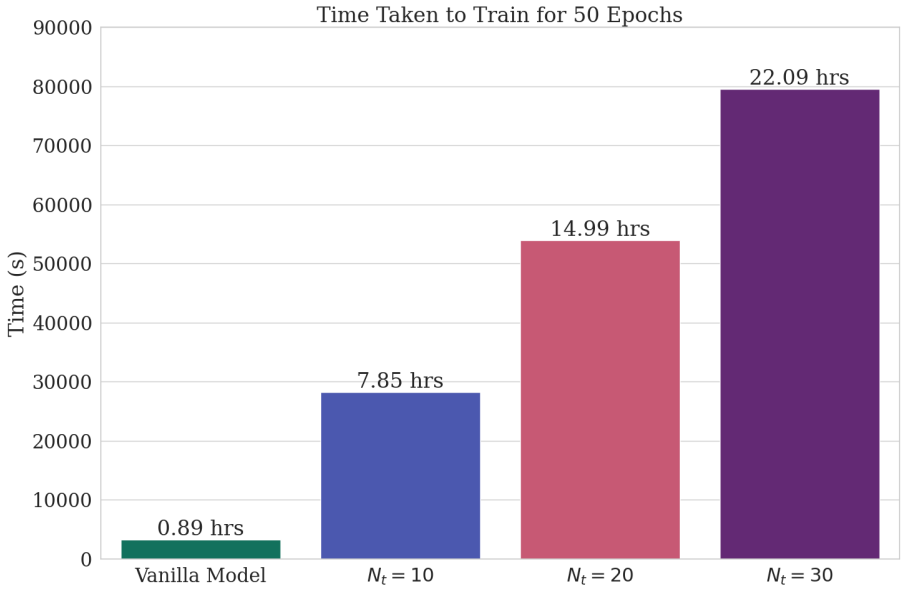


Figure 14: 50-epoch train times for the models on CelebA. Each loop was constrained to use the same GPU RAM.

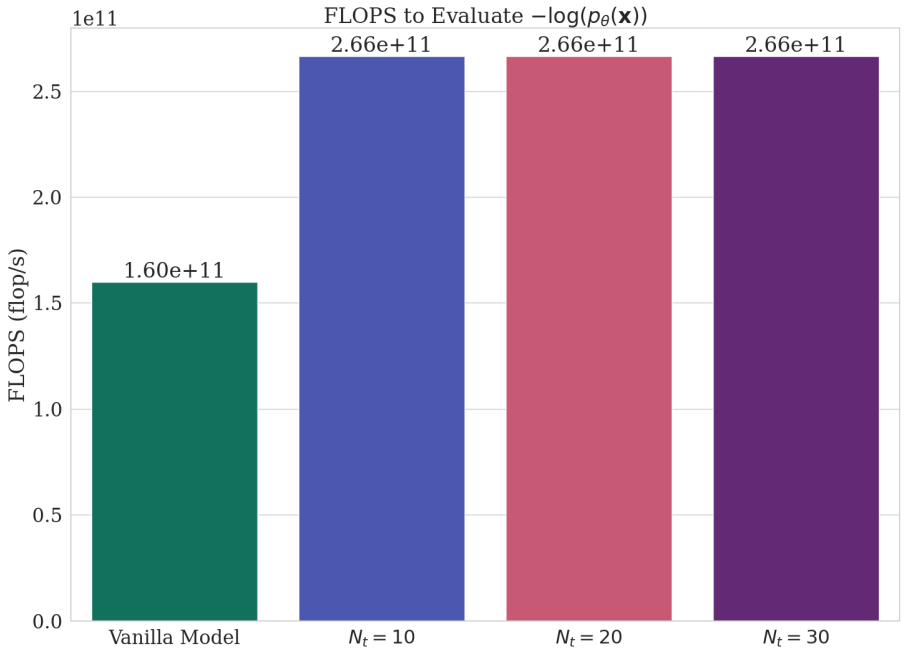


Figure 15: Floating point operations per second (FLOPs) required for the loss evaluation. FLOPs were recorded using JAX's xla_computation function. Despite apparently similar FLOPs for models with varying N_t, training times differ (Fig. 14). This discrepancy may be attributed to limitations regarding the estimates provided by the FLOPs profiler.

7.7.6 CIFAR-10 results

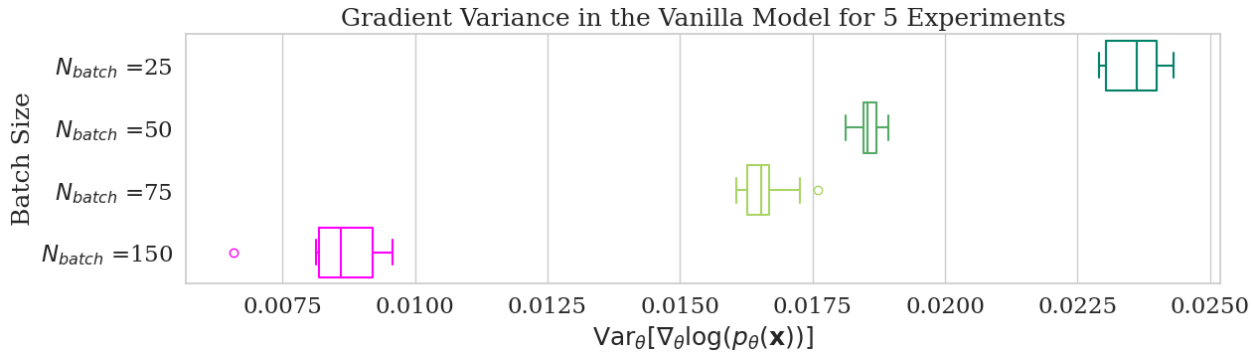


Figure 16: Relationship between final learning gradient variance of the vanilla model and batch size. The models were trained on CIFAR-10 for 50 epochs.

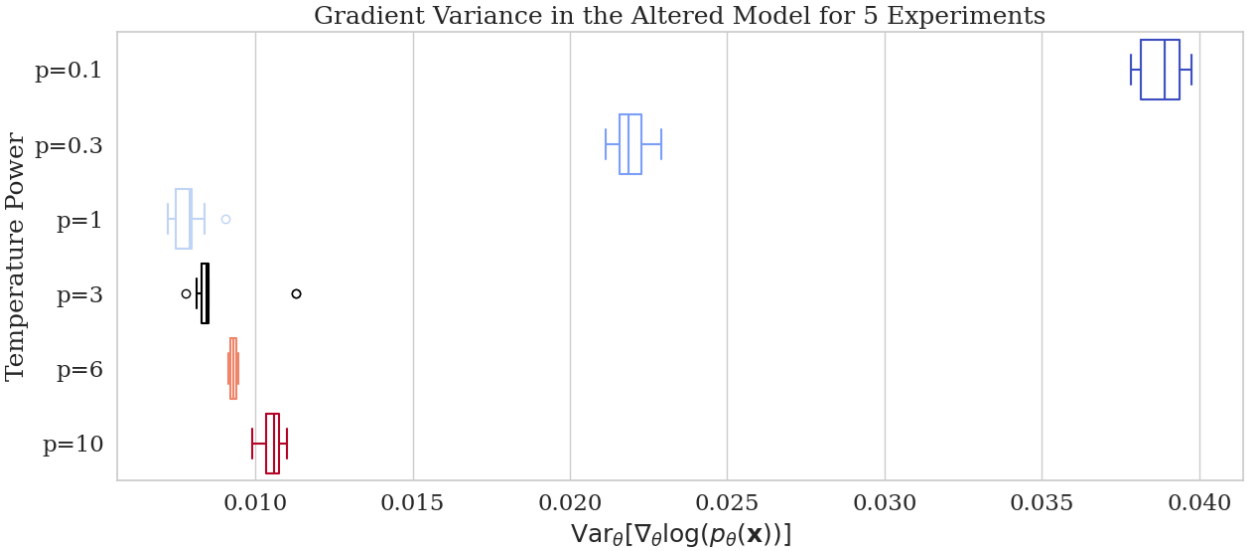


Figure 17: Relationship between final learning gradient variance and p in Eq. 21 the altered model. The models were once again trained on CIFAR-10 for 50 epochs. The same trends visible in Fig. 4 are apparent here.

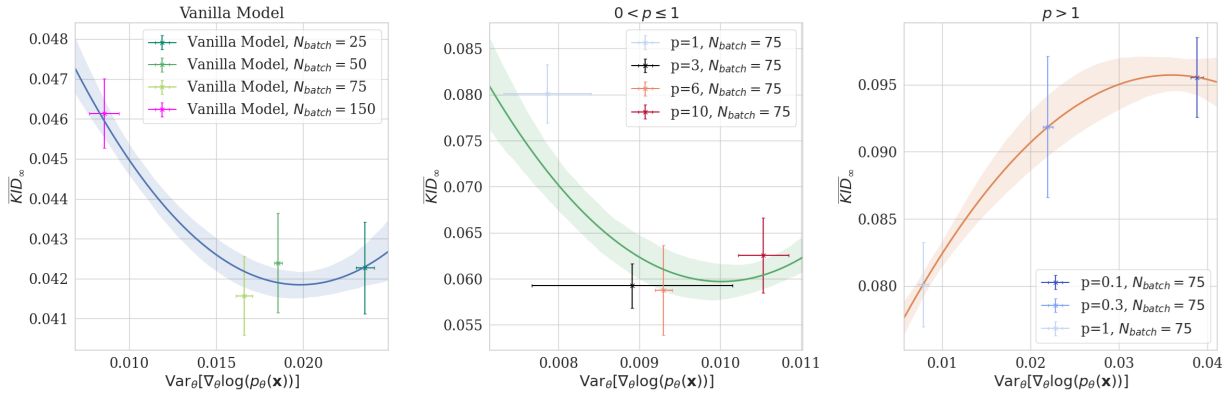


Figure 18: The relationships between learning gradient noise and generated image fidelity, as quantified by $\overline{KID}_\infty$ for CIFAR-10. This plot, equivalent to Fig. 19, has been separated into its different regimes for more clarity.

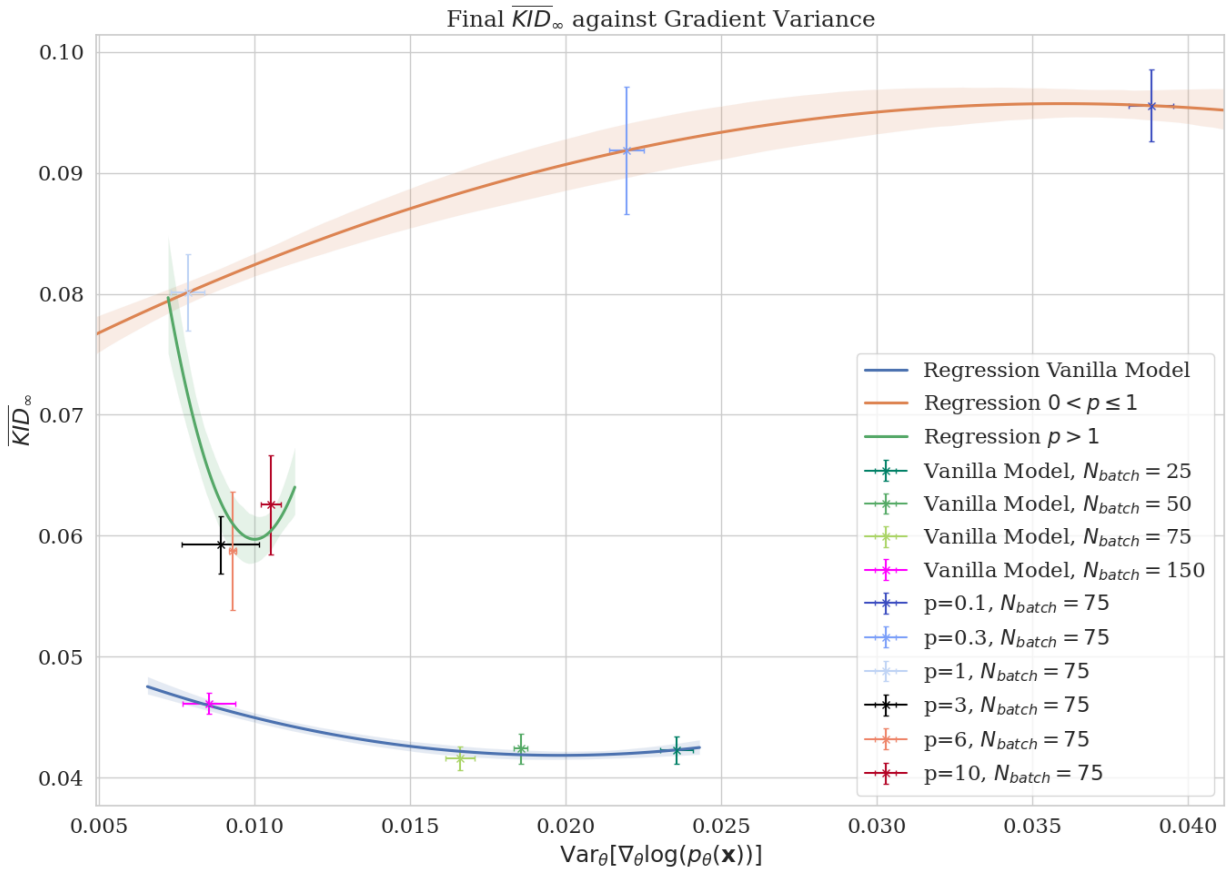


Figure 19: Learning gradient variance and $\overline{KID}_\infty$ for the models trained on CIFAR-10. The trends mirror that of Fig. 5, although the discrepancy between the vanilla and altered models has seemingly worsened and become more exaggerated.

(a) Vanilla Model $N_{batch} = 75, K_{\mathbf{z}|\mathbf{x}} = 40$

(b) $p = 0.1, N_t = 10, K_{\mathbf{z}|\mathbf{x},t} = 40$

(c) $p = 1, N_t = 10, K_{\mathbf{z}|\mathbf{x},t} = 40$

(d) $p = 6, N_t = 10, K_{\mathbf{z}|\mathbf{x},t} = 40$

Figure 20: Final CIFAR-10 images generated by the models depicted in Fig. 19. The discrepancy described in Section 5 between the altered and vanilla models has become more pronounced in the case of CIFAR-10.