Principled Gradient-Based MCMC for Conditional Sampling of Text

Li Du [1] **Afra Amini** [2] **Lucas Torroba Hennigen** [3] **Xinyan Velocity Yu** [4]
Holden Lee [1] **Jason Eisner** [1] **Ryan Cotterell** [2]

Abstract

We consider the problem of sampling text from an energy-based model. This arises, for example, when sampling text from a neural language model subject to soft constraints. Although the target distribution is discrete, the internal computations of the energy function (given by the language model) are differentiable, so one would like to exploit gradient information within a method such as MCMC. Alas, all previous attempts to generalize gradient-based MCMC to text sampling fail to sample correctly from the target distribution. We propose a solution, along with variants, and study its theoretical properties. Through experiments on various forms of text generation, we demonstrate that our unbiased samplers are able to generate more fluent text while better adhering to the control objectives. The same methods could be used to sample from discrete energy-based models unrelated to text.

1. Introduction

Recent papers have performed controlled text generation from pretrained language models by formulating energy-based models over text and applying Markov Chain Monte Carlo (MCMC) algorithms (Qin et al., 2022; Kumar et al., 2022; Mireshghallah et al., 2022; Amini et al., 2023). Energy-based language modeling is versatile, allowing a generic pretrained language model to be modified by arbitrary energy terms that express desired traits for the output text. The normalization constant is intractable to compute (Lin et al., 2021), as for other energy-based models (EBMs), but one can use MCMC algorithms to draw samples.

However, simple approaches to discrete MCMC, such as Gibbs sampling, tend to scale poorly (Deng et al., 2020), for reasons we discuss in §3.1. To address this problem, the approaches above exploit the fact that the original language model and the auxiliary energy terms are differentiable with respect to the continuous *embeddings* of the input tokens, even though the tokens themselves are discrete. The resulting first-order gradient information can be incorporated into an MCMC procedure, potentially accelerating convergence.

As one of the most successful gradient-based samplers, Hamiltonian Monte Carlo (HMC) and its variants (Duane et al., 1987; Neal, 1993; Hoffman and Gelman, 2014) have been proven to be highly effective in sampling from high-dimensional, continuous distributions, making them the default samplers of many probabilistic programming languages (Carpenter et al., 2017; Bingham et al., 2018; Phan et al., 2019). Adapting HMC to a discrete setting, Amini et al. (2023) recently proposed a promising sampler for controlled text generation. Alternatively, Langevin dynamics (Grenander and Miller, 1994; Welling and Teh, 2011), another gradient-based sampler, has been a more popular candidate to adapt into NLP models due to its simplicity.[1] As a result, Qin et al. (2022) and Kumar et al. (2022) proposed text samplers inspired by Langevin dynamics.

Unfortunately, a closer look reveals that *none* of these gradient-based Markov chains for text generation provably converge to their intended distributions in the limit, as we theoretically and empirically show in §3.3. While these previous papers did achieve good results on downstream tasks, this observation raises the question: What would happen if we sampled from the target distribution correctly?

In this work, we tackle this question by proposing several tractable gradient-based samplers that are *faithful* to the target energy-based text distribution, meaning that they have the correct limit distribution. We derive two novel samplers, based on Langevin dynamics and Gibbs sampling, respectively, and then develop their adaptive and hybrid variants. When applicable, we also prove convergence and mixing properties of our proposed samplers.

Faithful samplers are not guaranteed to outperform unfaith-

[1]Johns Hopkins University [2]ETH Zürich [3]MIT CSAIL [4]University of Southern California. Correspondence to: Li Du <leodu@cs.jhu.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

[1]In fact, it is well-known that Langevin dynamics can be seen as HMC where the Hamiltonian dynamics are simulated for a single step. See, e.g., Neal (1993) or Kennedy (1990).

ful methods empirically. Even though prior methods approach a biased version of the target distribution, they might benefit from approaching it more quickly, or the bias might actually be a useful adjustment to an imperfect energy function. Thus, through experiments on various forms of text generation, we explore whether our proposed faithful samplers can generate more fluent text while adhering to the control target better. In our experiments, our faithful samplers did outperform previous methods.

2. Energy-based Models of Text

Pretrained language models (Radford et al., 2019; Raffel et al., 2020; Brown et al., 2020) have demonstrated an impressive ability to generate fluent text. They do so by factorizing a locally normalized string-valued distribution $p_{\text{LM}}(\boldsymbol{w})$ ($\boldsymbol{w} \in \Sigma^*$) over some vocabulary Σ (Du et al., 2023):

$$p_{\text{LM}}(\boldsymbol{w}) = p_{\text{LM}}(\text{EOS} \mid \boldsymbol{w}) \prod_{n=1}^{N} p_{\text{LM}}(w_n \mid \boldsymbol{w}_{<n}) \quad (1)$$

They train the local conditionals $p_{\text{LM}}(\cdot \mid \boldsymbol{w}_{<n})$ on a massive text corpus. Such a corpus is often heterogeneous (derived from a mix of newspapers, blog posts, etc.) and does not focus on any particular topic, style, sentiment, or communicative intent. Language models are therefore often prompted with a prefix that influences the generation of subsequent words. For more precise control, it can be useful to employ *controlled generation*—sampling a text that satisfies one or several *explicit* soft constraints provided at runtime to the sampler. Such constraints can be lexical, semantic, grammatical, or arbitrary functions that evaluate some global property over the entire sequence. We decrease the (log-)probability of each text to the degree that it violates the soft constraints. This leaves us with the challenging problem of sampling from a (discrete) unnormalized probability distribution, which may be presented in the form of an energy-based model (EBM; Hinton 2002; LeCun et al. 2007):

$$\pi(\boldsymbol{w}) = \frac{1}{Z} \exp(-U(\boldsymbol{w})) \quad (2)$$

Here $U(\boldsymbol{w})$ is called the *energy function*.[2] The flexibility of this framework lies in the fact that one can refine an existing model by coupling its energy function with arbitrary soft constraint functions that assess whether the output text has desirable attributes. Concretely, we can set

$$U(\boldsymbol{w}) = U_{\text{LM}}(\boldsymbol{w}) + \sum_{i=1}^{I} U_i(\boldsymbol{w}) \quad (3)$$

where $U_{\text{LM}}(\boldsymbol{w}) \stackrel{\text{def}}{=} -\log p_{\text{LM}}(\boldsymbol{w})$ (from Eq. (1)) and each of the I constraint functions $U_i(\boldsymbol{w})$ measures the extent to which the sequence $\boldsymbol{w}$ violates the i^{th} constraint. This energy function yields a distribution that is related to $p_{\text{LM}}(\boldsymbol{w})$ but places more probability mass on the sequences that better satisfy the constraints.

3. Text Generation as MCMC

3.1. Sampling from EBMs

The flexible formulation in Eqs. (2) and (3) allows us to cast controlled text generation as the problem of sampling from an energy-based model. However, EBMs can be challenging to sample from.

Consider sampling a sequence of N words $\boldsymbol{w} = w_1 \cdots w_N \in \Sigma^N$ from the EBM defined by Eqs. (2) and (3). The normalization constant Z from this EBM is an intractable sum of $|\Sigma|^N$ terms.[3] The locally normalized conditional probabilities needed for left-to-right autoregressive sampling are effectively ratios of such normalization constants, and are likewise intractable (Lin et al., 2021).

As for other unnormalized distributions, we may resort to designing a Markov Chain Monte Carlo (MCMC) sampler. In our situation, the combinatorially large underlying state space Σ^N means that the basic Random Walk Metropolis algorithm (Metropolis et al., 1953) would have near-zero acceptance rate: most uniform samples from Σ^N are improbable under π. Gibbs sampling (Geman and Geman, 1984), another commonly used MCMC algorithm, requires one to be able to efficiently sample from the conditional $\pi(w'_n \mid \boldsymbol{w}_{\backslash n})$.[4] This is also impractical since sampling $\pi(\cdot \mid \boldsymbol{w}_{\backslash n})$ in a locally normalized LM would be slow.[5]

3.2. Gradient-based Sampling via Continuous Relaxation

The challenges outlined in the previous section indicates that we need additional techniques to obtain a sampling procedure that yields quality samples in a reasonable amount of time. Observing that U_{LM} defined from a pretrained neural LM is differentiable, as well as possibly the constraint func-

[2] The notation $U(\boldsymbol{w})$ rather than the usual $E(\boldsymbol{x})$ is drawn from the HMC literature, which calls it the *potential function*. Energy-based models (2) are sometimes trained directly, for example by noise-contrastive estimation (Deng et al., 2020); our sampler would work for these models as well as for the model in Eq. (3).

[3] It can be tractable in special cases such as a linear-chain Markov random field, but is not in general.

[4] We use $\boldsymbol{w}_{\backslash n}$ to denote the set of random variables of all indices *except* n, i.e., $\boldsymbol{w}_{\backslash n} = w_1 \cdots w_{n-1} w_{n+1} \cdots w_N$.

[5] Doing so would involve computing $U(\boldsymbol{w}')$ for $|\Sigma| - 1$ strings $\boldsymbol{w}'$ obtained by replacing w_n in different ways. When U_{LM} is autoregressive, each $U(\boldsymbol{w}')$ takes time $\Omega(N - (n-1))$ to compute. A more practical alternative is the Metropolis-within-Gibbs technique of sampling w'_n from some faster proposal distribution, subject to a Metropolis–Hastings acceptance probability that considers $U(\boldsymbol{w}')$ for only the single proposed $\boldsymbol{w}'$. We will apply that technique ourselves in §4.3, using a novel gradient-based proposal distribution.

tions U_i, several prior work have taken inspiration from the success of gradient-based sampling in other domains (Neal, 2011; Hoffman and Gelman, 2014; Carpenter et al., 2017; Welling and Teh, 2011; Du and Mordatch, 2019; Song and Ermon, 2020) and attempted to leverage gradient information when sampling from text-based EBMs (Qin et al., 2022; Kumar et al., 2022; Amini et al., 2023).

However, problems arise because gradient-based sampling algorithms such as HMC or Langevin dynamics only directly apply to continuous distributions. To apply such algorithms to sample from discrete distributions, prior work that developed gradient-based sampling for energy-based text generation all focus on continuous relaxations of the underlying discrete space. In particular, Qin et al. (2022) runs a sampler entirely in the continuous space $\mathbb{R}^d$ and then finds a discrete word sequence with similar embeddings. Kumar et al. (2022) similarly take gradient-based random steps in $\mathbb{R}^d$ but project back to a discrete word sequence after *each* step. Amini et al. (2023) use Voronoi tessellation to relax the discrete distribution over word embeddings into a piecewise continuous distribution with the embeddings as the centers of the Voronoi cells.

Unfortunately—as we show below—none of these continuous relaxation techniques resulted in a sampler that correctly samples from the target energy-based distribution over text. Moreover, correcting these samplers with the Metropolis-Hastings technique (App. C.3) is not possible in practice because the required acceptance probability is intractable. This is because while Kumar et al. (2022) or Amini et al. (2023) can *sample* from their transition kernel $q(\boldsymbol{w}' \mid \boldsymbol{w})$, the *probability* of this discrete transition (and of the reverse transition) cannot be computed in closed form, as it involves an integral over an intermediate continuous draw (App. B.1).

3.3. Unfaithfulness of Gradient-based Text Samplers

In this section, we explain and illustrate in detail why existing methods fail to converge to their intended distributions and thus are unfaithful samplers. To do so, we consider the setting of sampling a sequence of N words $\boldsymbol{w} = w_1 \cdots w_N \in \Sigma^N$ from an energy-based sequence model. With a slight abuse of notation, we assume that $U(\boldsymbol{w})$ takes the form $U(\boldsymbol{x})$ where $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N) \in \mathcal{X} \stackrel{\text{def}}{=} \mathcal{V}^N \subset \mathbb{R}^{Nd}$ is the sequence of word embeddings from the finite set $\mathcal{V} \subset \mathbb{R}^d$.

COLD (Qin et al., 2022). COLD observes that, while the EBM induced from a language model is defined as

$$\pi_{\text{LM}}(\boldsymbol{x}) = \frac{\exp(-U_{\text{LM}}(\boldsymbol{x}))}{\sum_{\boldsymbol{y} \in \mathcal{X}} \exp(-U_{\text{LM}}(\boldsymbol{y}))}\,,\ \boldsymbol{x} \in \mathcal{X} \tag{4}$$

the implementation of the energy function $U(\boldsymbol{x})$ can also take vectors other than word embeddings as its input. COLD proceeds to use Langevin dynamics that include $U(\boldsymbol{x})$ as an energy function over the continuously relaxed space. In effect, COLD samples from a *density* similar to[6]

$$\tilde{\pi}_{\text{COLD-LIKE}}(\boldsymbol{x}) = \frac{\exp(-U_{\text{LM}}(\boldsymbol{x}))}{\displaystyle\int_{\mathbb{R}^{Nd}} \exp(-U_{\text{LM}}(\boldsymbol{y}))\,\mathrm{d}\boldsymbol{y}}\,,\ \boldsymbol{x} \in \mathbb{R}^{Nd} \tag{5}$$

which has the same numerator as Eq. (4). It then generates a discrete sentence $\boldsymbol{w}$ from left to right, using a rounding heuristic that tries to remain plausible under $p_{\text{LM}}(\boldsymbol{w})$ while using word embeddings similar to the sampled $\boldsymbol{x} \in \mathbb{R}^{Nd}$.

When COLD performs Langevin dynamics over Eq. (5), its samples are not distributed according to Eq. (4). We will illustrate this formally in Example 3.1 below.

As a simple illustration of the problem, suppose that $N = 1$ and that the words *water* and *beer* have equal energies and thus have equal probabilities under the target discrete EBM. COLD is about equally likely to draw $\boldsymbol{x}_1$ from an ϵ-ball around each word. However, suppose there are more words in the vicinity of *beer*: the ϵ-ball around *beer* also includes several slang synonyms (*hooch*, *booze*, etc.). Then drawing $\boldsymbol{x}_1$ in that ϵ-ball may result in rounding to one of these near neighbors instead of *beer*. So the probability that COLD specifically samples *beer* may—incorrectly—be several times less than the probability that it samples *water*.

MUCOLA (Kumar et al., 2022). Similar to COLD, MUCOLA also takes Langevin steps in the underlying continuous space $\mathbb{R}^{Nd}$, but after each step, it projects back to the nearest point in the discrete embedding space $\mathcal{X}$:

$$\boldsymbol{x}' = \text{Proj}_{\mathcal{X}}\left(\boldsymbol{x} - \frac{\alpha}{2}\nabla U(\boldsymbol{x}) + \sqrt{\alpha}\boldsymbol{\xi}\right) \tag{6}$$

where $\boldsymbol{\xi} \sim \mathcal{N}(0, I)$, α is the stepsize, and $\text{Proj}_{\mathcal{X}}$ uses the Euclidean metric.

Troublingly, the update equation (6) and hence the stationary distribution of MUCOLA depends only on the gradients $\nabla U(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{X}$, and not on the actual values $U(\boldsymbol{x})$. Thus, given an energy function U, it is easy to construct infinitely many other energy functions U' on which MUCOLA would have the same behavior, but which all give rise to different EBMs. Clearly, MUCOLA samples correctly from at most one of these EBMs.

This shows that MUCOLA is unfaithful. In Example 3.1 below, we give a concrete example where Eq. (6) fails to converge to the target distribution.

SVS (Amini et al., 2023). SVS resembles COLD in that it correctly samples $\boldsymbol{x}$ from some continuous energy-based

[6] In practice, COLD operates in logit space and uses a weighted average of word embeddings, which keeps it in the convex hull of $\mathcal{V}^N$ and ensures a finite denominator. However, this detail does not affect our following point.

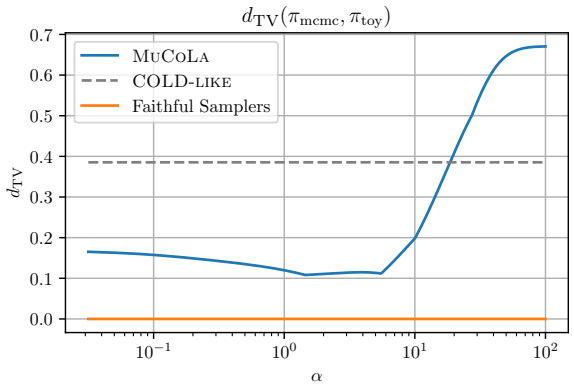

Figure 1. Total variation distance between π_{mcmc}, the limiting distribution of MCMC algorithms from previous work, and π_{toy}, the toy language model distribution from Example 3.1. π_{mcmc} is computed with spectral decomposition when possible. We can observe that the limiting distribution of COLD is far from the target distribution, and MuCoLa, depending on its step size α, may be close to the target distribution. Nevertheless, it does not have the correct distribution for any α.

model over $\mathbb{R}^{Nd}$ (though it uses HMC to do so) and then rounds to obtain $\boldsymbol{w}$. The main difference is that it attempts to construct this continuous EBM so that the full procedure actually samples from the target distribution. Specifically, it constructs a "relaxation" of the original EBM: a piecewise Gaussian energy function in which the Voronoi cell around each $\boldsymbol{x} \in \mathcal{X}$ has a truncated Gaussian, centered at $\boldsymbol{x}$ and scaled to have the correct integral. Unfortunately, finding the correct scaling factors would require actually computing high-dimensional Gaussian integrals, which is infeasible App. B. Thus, the SVS implementation drops the scaling factors, which is only correct for simple symmetric models such as Ising models.

Example 3.1 (A Toy Energy-based LM)**.** To further illustrate the previous claims, we consider a toy energy-based LM over a sequence of N tokens, with a binary vocabulary and a one-dimensional embedding $\mathcal{V} = \Sigma = \{-1, +1\}$. The energy function we use has the form

$$U(\boldsymbol{x}) = -\beta(\tfrac{1}{2}\boldsymbol{x}^\top A \boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x}) \tag{7}$$

with $\boldsymbol{x} \in \mathcal{V}^N$ and $\pi_{\mathrm{toy}}(\boldsymbol{x}) \propto \exp(-U(\boldsymbol{x}))$. Concretely, we set A to be the adjacency matrix of an N-cycle and $\boldsymbol{b} = \boldsymbol{0}$. This energy-based LM is a so-called linear-chain Ising model with zero magnetic field.

We choose this model for the following reasons:

1. The energy function is differentiable, and hence all previous algorithms apply;
2. When N is not too large, we can compute the exact distribution;
3. The binary vocabulary allows us to compute the transition matrix of MuCoLa exactly as well as its stationary distribution.[7]

We set $N = 5$ and use spectral decomposition of the transition matrix to calculate the exact stationary distribution of MuCoLa. For COLD, we estimate the multi-dimensional Gaussian's quadrant probabilities with 1 million samples.

From Fig. 1, we can see that the limiting distribution of COLD fails to match the target language model distribution π_{toy}, as we remarked earlier. On the other hand, we interestingly observe that, for a certain range of α, MuCoLa can in fact approximate the true distribution fairly well. This may explain the fact that MuCoLa performs better than COLD in actual language generation tasks. Nevertheless, MuCoLa is not able to sample from the true distribution regardless of the value of α.[8] //

4. Faithful Gradient-based Text Generation

In this section, we develop faithful samplers. We first develop a Langevin-based sampler in §4.1, which we term p-NCG. We discuss its theoretical properties in §4.2. We then develop a Gibbs-based sampler in §4.3. We conclude with a discussion on hybrid samplers in §4.4.

We accomplish this by returning to the standard Metropolis–Hastings scheme (reviewed in App. C.3). While we again use a gradient-informed transition kernel, we construct it such that the discrete transition probabilities $p(\boldsymbol{x}' \mid \boldsymbol{x})$ and $p(\boldsymbol{x} \mid \boldsymbol{x}')$ can be computed in closed form. This allows us to compute the acceptance probability of a proposed transition.

4.1. A Langevin-based Sampler

Given the Langevin update in continuous space,

$$\boldsymbol{x}' = \boldsymbol{x} - \frac{\alpha}{2}\nabla U(\boldsymbol{x}) + \sqrt{\alpha}\,\boldsymbol{\xi} \tag{8}$$

where $\boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{0}, I)$, a natural way to adapt it to our discrete setting is to project the updated coordinates in the relaxed continuous space back to the discrete space $\mathcal{X}$ of word sequence embeddings:

$$\boldsymbol{x}' = \mathrm{Proj}_{\mathcal{X}}\Big(\underbrace{\boldsymbol{x} - \frac{\alpha}{2}\nabla U(\boldsymbol{x})}_{\boldsymbol{\mu}_{\boldsymbol{x}}} + \sqrt{\alpha}\,\boldsymbol{\xi}\Big) \tag{9}$$

This strategy is used by MuCoLa (Kumar et al., 2022). It works reasonably well[9] but is biased as discussed in §3.3. Moreover, it cannot be corrected by Metropolis–Hastings

[7] The transition probability of MuCoLa is in general infeasible to compute. See App. B.1.

[8] We also note that, in this specific model, SVS is able to sample from the correct distribution because the Voronoi cells induced by the embeddings have equal measure due to symmetry. However, this is not true in general language models.

[9] In our preliminary experiments, we found MuCoLa to be the best-performing previous method.

because the acceptance probability involves computing the same high-dimensional integral that made the HMC-based sampler in SVS (Amini et al., 2023) infeasible.[10]

The key property of Eq. (9) is that it is likely to sample $\boldsymbol{x}' \in \mathcal{X}$ that is close to $\boldsymbol{\mu}_{\boldsymbol{x}}$. After all, if we omitted the projection operator, then we would have $\boldsymbol{x}' = \boldsymbol{\mu}_{\boldsymbol{x}} + \sqrt{\alpha}\,\boldsymbol{\xi}$. That is, the probability of drawing a specific $\boldsymbol{x}' \in \mathbb{R}^{Nd}$ would simply be the probability of drawing $\boldsymbol{x}'$ from $\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{x}}, \alpha)$. This probability decreases as $\boldsymbol{x}'$ moves farther away from $\boldsymbol{\mu}_{\boldsymbol{x}}$, which also tends to be true after projection.

We can preserve this property of MuCoLa without using the projection operator. Instead, we directly define a discrete proposal distribution $q(\boldsymbol{x}' \mid \boldsymbol{x})$, by choosing $\boldsymbol{x}' \in \mathcal{X}$ with probability proportional to its density under $\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{x}}, \alpha)$:

$$q(\boldsymbol{x}' \mid \boldsymbol{x}) \propto \exp\left(-\frac{\|\boldsymbol{x}' - \boldsymbol{\mu}_{\boldsymbol{x}}\|_2^2}{2\alpha}\right) \tag{10a}$$

$$= \exp\left(-\frac{1}{2\alpha}\left\|\boldsymbol{x}' - \left(\boldsymbol{x} - \frac{\alpha}{2}\nabla U(\boldsymbol{x})\right)\right\|_2^2\right) \tag{10b}$$

The Metropolis–Hastings acceptance probability associated with this proposal distribution is simple to compute. Because there is no projection operator, we avoid the infeasible integral that would be needed to correct MuCoLa with Metropolis–Hastings.

This variant also avoids the *water/beer* problem that plagued all the methods of §3.3. Our proposal distribution $q(\boldsymbol{x}' \mid \boldsymbol{x})$ does not care whether $\boldsymbol{x}'$ has many near neighbors in the discrete space $\mathcal{X}$. Since our target distribution $\pi(\boldsymbol{x}')$ in Eq. (2) does not care either, our proposal distribution is well-matched to the target distribution and should enjoy a high acceptance rate. For example, in a symmetric situation where $U(\textit{water}) = U(\textit{beer})$ (so $\pi(\textit{water}) = \pi(\textit{beer})$) and $||\textit{beer} - \boldsymbol{\mu}_{\textit{water}}|| = ||\textit{water} - \boldsymbol{\mu}_{\textit{beer}}||$ (so $q(\textit{beer} \mid \textit{water}) = q(\textit{water} \mid \textit{beer})$), the Metropolis–Hastings acceptance probability (28) is 1, even if *beer* has more near neighbors. In contrast, MuCoLa would have $q(\textit{beer} \mid \textit{water}) < q(\textit{water} \mid \textit{beer})$ if *beer* has more near neighbors, since then there is a low probability of choosing a step $\sqrt{\alpha}\,\boldsymbol{\xi}$ that comes closer to *beer* than to any of its neighbors.

With a few steps of derivation (detailed in App. D), we can rewrite the proposal in Eq. (10b) as

$$q(\boldsymbol{x}' \mid \boldsymbol{x}) \propto \exp\Bigg(\underbrace{-\frac{1}{2}\nabla U(\boldsymbol{x})^\top(\boldsymbol{x}' - \boldsymbol{x})}_{\text{Term (A1)}} \underbrace{-\frac{1}{2\alpha}\|\boldsymbol{x}' - \boldsymbol{x}\|_2^2}_{\text{Term (A2)}}\Bigg) \tag{11}$$

Let us examine Eq. (11) more closely. We notice that Term (A1) is in effect performing a first-order Taylor expansion, i.e., $U(\boldsymbol{x}') - U(\boldsymbol{x}) \approx \nabla U(\boldsymbol{x})^\top(\boldsymbol{x}' - \boldsymbol{x})$, in an attempt to move to a state with lower energy. On the other hand, first-order approximation is only accurate locally, and hence Term (A2) acts as a regularizer that decreases the probability of moving to $\boldsymbol{x}'$ that is too far from $\boldsymbol{x}$. The regularizer is stronger for small stepsize α.

Finally, when applying Eq. (11) to realistic language models such as GPT-2, we found that the ℓ_2-norm penalty often runs into pathological situations where a few indices' large deviation disrupts the proposal distribution and results in low acceptance rate. We hypothesize that this is due to the unusual geometry of the underlying embedding space (Mimno and Thompson, 2017) and found that using alternative norms is an effective remedy. This leads to our final form of proposal distribution:

$$q(\boldsymbol{x}' \mid \boldsymbol{x}) \propto \exp\Bigg(\underbrace{-\frac{1}{2}\nabla U(\boldsymbol{x})^\top(\boldsymbol{x}' - \boldsymbol{x})}_{\text{Term (B1)}} \underbrace{-\frac{1}{2\alpha}\|\boldsymbol{x}' - \boldsymbol{x}\|_p^p}_{\text{Term (B2)}}\Bigg) \tag{12}$$

We call this method ℓ_p-Norm Constrained Gradient sampler (p-**NCG**), due to its connection to the Norm Constrained Gradient sampler proposed in Rhodes and Gutmann (2022), which is the special case of $p = 2$ as in Eq. (11). The NCG sampler is also referred to as R-MALA in Grathwohl et al. (2021, Eq. 24) and D-MALA in Zhang et al. (2022); see App. A for more details. Specifically, Zhang et al. (2022) extensively studied many of the interesting properties of D-MALA, which we build on in the next section.

4.2. Properties of p-NCG

Independence of Positions. Suppose we are sampling a sequence of length N using the word embeddings: $\boldsymbol{x} = [\boldsymbol{x}_1 \cdots \boldsymbol{x}_N] \in \mathbb{R}^{Nh}$ where each $\boldsymbol{x}_n \in \mathcal{X} \subset \mathbb{R}^h$ is a word embedding. The proposal distribution in Eq. (12) factorizes as a product over the N positions:

$$q(\boldsymbol{x}' \mid \boldsymbol{x}) = \prod_{n=1}^{N} q(\boldsymbol{x}'_n \mid \boldsymbol{x}) \quad \text{(conditional independence)}$$

$$q(\boldsymbol{x}'_n \mid \boldsymbol{x}) \propto \exp\left(-\frac{1}{2}\nabla_n U(\boldsymbol{x})^\top(\boldsymbol{x}'_n - \boldsymbol{x}_n) - \frac{1}{2\alpha}\|\boldsymbol{x}'_n - \boldsymbol{x}_n\|_p^p\right) \tag{13}$$

This means that we can sample all of the word positions in parallel, with a separate softmax over the vocabulary $|\Sigma|$ at each position. It is not necessary to normalize by brute force over all $|\Sigma|^N$ word sequences in $\mathcal{X}$.

Convergence Analysis. Another interesting property of the p-NCG proposal is that when used unadjusted[11] on dis-

[10] For details of why this integral shows up in metropolizing MuCoLa, see App. B.1 for details.

[11] As is standard in MCMC literature, we say that a proposal is used *unadjusted* if we omit the Metropolis-Hastings correction

crete log-quadratic distributions, such as the Ising models, its stationary distribution converges to the target distribution as the step size α tends to zero.

Definition 4.1. *Let $\pi(\boldsymbol{x})$ be a discrete distribution over $\mathcal{X} \subset \mathbb{R}^d$ where $|\mathcal{X}| < \infty$. π is* log-quadratic *if it can be expressed as*

$$\pi(\boldsymbol{x}) \propto \exp\left(\boldsymbol{x}^\top A \boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x}\right) \tag{14}$$

for some $A \in \mathbb{R}^{d\times d}$ and $\boldsymbol{b} \in \mathbb{R}^d$.

Theorem 4.2. *Let $\pi(\boldsymbol{x})$ be a discrete log-quadratic distribution as defined in Def. 4.1. For any $\alpha > 0$, there exists a unique distribution $\pi_\alpha(\boldsymbol{x})$ such that the Markov chain defined by q in Eq. (12) is reversible with respect to π_α and thus π_α is its stationary distribution. Further, $\pi_\alpha \to \pi$ weakly as $\alpha \to 0$.*

Proof Idea. The key insight of the proof is that first-order approximation of a quadratic energy function will leave a symmetric second-order error term. One can exploit this symmetry to explicitly construct a reversing distribution π_α. One can then show that for this specific distribution, $\pi_\alpha \to \pi$ as claimed. See App. E for the full proof. □

More generally, Theorem 4.2 shows that the p-NCG is *locally-balanced* with respect to discrete log-quadratic distributions. Introduced in Zanella (2020, §2.2), a proposal is said to be locally-balanced with respect to the target distribution if its unadjusted limiting distribution converges weakly to the target distribution. Many recent works have found that being locally-balanced is a favorable property of a proposal distribution (Zanella, 2020; Grathwohl et al., 2021; Sun et al., 2022, *inter alia*).

Mixing-time Analysis. When unadjusted proposals exhibit limiting behaviors as in Theorem 4.2, it is tempting to use the proposal without using Metropolis–Hastings correction, as argued in Zhang et al. (2022). However, as Theorem 4.3 shows, the mixing time (defined in App. C.4) increases exponentially as the step size decreases towards 0. This means that, in practice, using the unadjusted proposal with a small step size is infeasible.

Theorem 4.3. *Let $\pi(\boldsymbol{x})$ be a discrete log-quadratic distribution as defined in Def. 4.1. There exist constants $c_1, c_2, Z > 0$ that depends only on $\pi(\boldsymbol{x})$ such that the mixing time of q in Eq. (12) satisfies*

$$t_{\text{mix}}(\varepsilon) \geq \left(\frac{c_1}{Z}\exp\left(\frac{c_2}{2\alpha}\right) - 1\right)\log\left(\frac{1}{2\varepsilon}\right) \tag{15}$$

Proof Idea. We use the Geršgorin disc theorem (Theorem F.1) to bound the location of the eigenvalues and then relate it to mixing time through a well-known inequality (Theorem F.2). See App. F for the full proof. □

and accept every sample.

4.3. A Gibbs-based Sampler

In this section, we adapt the Gibbs sampler (Geman and Geman, 1984). Again, consider sampling a sequence of length N with word embeddings $\boldsymbol{x} = [\boldsymbol{x}_1 \cdots \boldsymbol{x}_N] \in \mathbb{R}^{Nh}$ where each $\boldsymbol{x}_n \in \mathcal{X} \subset \mathbb{R}^h$ is a word embedding. To be able to use Gibbs sampling, we need to be able to efficiently compute the conditional probabilities $\pi(\boldsymbol{x}_n \mid \boldsymbol{x}_{\setminus n})$, which is uncomfortably expensive as we argued in §3.1.

However, we recall the fact that Gibbs sampling is a special case of Metropolis–Hastings, where the use of exact conditional $\pi(\boldsymbol{x}_n \mid \boldsymbol{x}_{\setminus n})$ results in an acceptance probability of 1. We may instead use an approximation to $\pi(\boldsymbol{x}_n \mid \boldsymbol{x}_{\setminus n})$ and correct for the approximation error with Metropolis–Hastings. This is a instance of the method sometimes called *"Metropolis-within-Gibbs"*, which is well-known in the literature (Robert and Casella, 2004, §10.3.3, *inter alia*).

Specifically, taking advantage of gradient information as in §4.1, we approximate $\pi(\boldsymbol{x}_n \mid \boldsymbol{x}_{\setminus n})$ by estimating the energy difference with Taylor expansion:

$$\begin{aligned} U(\cdots, \hat{\boldsymbol{x}}_n, \cdots) - U(\cdots, \boldsymbol{x}_n, \cdots) \\ \approx \nabla_n U(\boldsymbol{x})^\top (\hat{\boldsymbol{x}}_n - \boldsymbol{x}_n) \end{aligned} \tag{16}$$

and then sample from

$$\exp(-\nabla_n U(\boldsymbol{x})^\top (\boldsymbol{x}'_n - \boldsymbol{x}_n)) \tag{17}$$

However, using the first-order approximation directly will lead to a near-zero acceptance rate due to the fact that local approximations have extremely high errors when used over the entire word embedding space. We therefore need to restrict the proposal move locally, which we again achieve by adding a p-norm penalty to our proposal. This yields a Gibbs-based proposal

$$\begin{aligned} q(\boldsymbol{x}'_n \mid \boldsymbol{x}_{\setminus n}) \propto \\ \exp\left(-\nabla_n U(\boldsymbol{x})^\top (\boldsymbol{x}'_n - \boldsymbol{x}_n) - \frac{1}{\alpha}\|\boldsymbol{x}'_n - \boldsymbol{x}_n\|_p^p\right) \end{aligned} \tag{18}$$

An important caveat is that, since we are already using the Metropolis–Hastings correction, it is a waste of computation to have self-transition probabilities in the proposal distribution.[12] This leads us to remove the self-transition probability and arrive at our final form of the Gibbs-based proposal:

$$q(\boldsymbol{x}'_n \mid \boldsymbol{x}_{\setminus n}) \propto \begin{cases} 0 & \text{when } \boldsymbol{x}_n = \boldsymbol{x}'_n \\ \text{Eq. (18)} & \text{otherwise} \end{cases} \tag{19}$$

[12] For example, the Metropolis sampler never proposes self-transitions, which is part of the reason for why it is known to mix faster than the standard Gibbs sampler (Glauber dynamics) on Ising models (MacKay, 2003, §31.1, p. 403) or other binary distributions (Newman and Barkema, 1999).

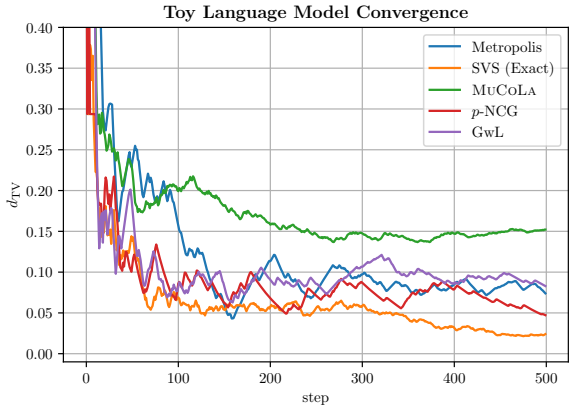

Figure 2. Total variation distance between the empirical distribution of different samplers (at different steps) and π_{toy}, the true distribution of the toy language model from Example 3.1.

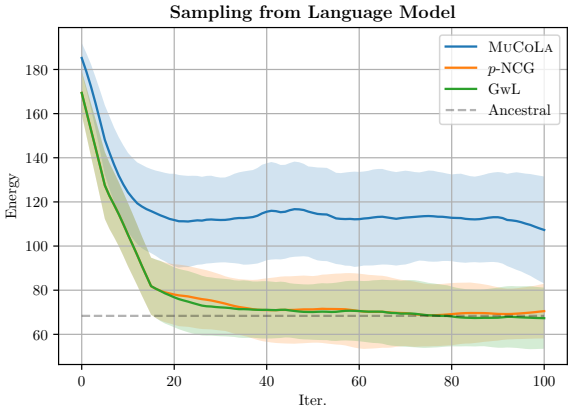

Figure 3. Energy of the Markov chain's state $\boldsymbol{x}$ over time as the Markov chain mixes (§5.2). Each color is a different sampler. The shaded region shows the middle 95% of energies at time t over 100 runs, while the solid line shows the mean. The dashed line is the mean energy of a random sample from the actual EBM π, namely GPT-2, estimated with 2000 samples.

Notice that Eq. (18) resembles Eq. (12) except for the factor $1/2$ and the single word update. For this reason, we call this sampler *Gibbs with Langevin* (GwL).

Scan Ordering. As with other Gibbs samplers, the scan ordering (the order in which each index is sampled) can greatly impact the sampler's efficiency (He et al., 2016).[13] In light of this, we will experiment with both systematic scan as well as random scan when using GwL.

4.4. Hybrid Samplers

Why would one use GwL when p-NCG can update multiple words at a time? We observed that when the sequence is randomly initialized, p-NCG indeed proposes to change multiple indices at once and can have a reasonably high acceptance rate. However, once the chain is close to convergence and the sentence structure starts to emerge, p-NCG only proposes to change at most 1 index at a time and proposes self-transitions roughly 15% of the time.[14] For this reason, GwL, which never proposes self-transitions, can have higher statistical efficiency in the later stages of the sampling process. In practice, we implement a hybrid sampler, where we use p-NCG during the initial phase of the sampler and switch to GwL once the chain starts to converge.

5. Unconditional Sampling Experiments

We first empirically assess the performance of our proposed samplers on unconditional sampling from EBMs. See App. G for full details of the experimental setup.

5.1. Toy Example

We first apply different sampling methods to the toy language model discussed in Example 3.1. Since we can exactly compute π_{toy} for small N, we can measure the total variation distance between the target distribution π_{toy} and the empirical distribution of the Markov chain at a given time step.

We compare our proposed samplers, p-NCG and GwL, to baselines in prior work, SVS and MuCoLa. As discussed in App. G.2, we set $p = 1$ for p-NCG, and tuned the step size α via grid search to minimize the average energy of our samples. We also included the standard Metropolis sampler (MacKay, 2003, §31) for comparison. Since SVS uses Gaussian augmentation, the resulting Hamiltonian yields a set of differential equations that can be solved in closed form. We therefore integrate the Hamiltonian dynamics exactly instead of using leapfrog steps, similar to the setup in Pakman and Paninski (2013).[15]

The results are shown in Fig. 2. We observe that the HMC-based sampler SVS has the best overall performance, due to its ability to traverse a long distance in the underlying space in a single step while preserving a perfect acceptance rate since its Hamiltonian is integrated exactly. It is important to note that SVS is *only* exact for simple symmetric distributions like the Ising model, as discussed in §3.3. It is biased in all realistic models (including all our subsequent experiments). On the other hand, p-NCG, GwL and Metropolis all

[13] This is despite the fact that systematic scan and random scan have long been conjectured to have similar mixing times up to logarithmic factors (Levin and Peres, 2017, §26, Open Question 3).

[14] That said, self-transitions are actually fast in wall-clock time beacause the energy and gradient can be reused on the next step.

[15] This exact integration is possible but can be difficult to implement when using SVS on actual language models. However, this is only an implementation speed up and does not alter the limiting distribution of SVS, which is incorrect for actual language models.

	Topic Control					Sentiment Control				
	Success(↑)	PPL(↓)	Dist-1(↑)	Dist-2(↑)	Dist-3(↑)	Success(↑)	PPL(↓)	Dist-1(↑)	Dist-2(↑)	Dist-3(↑)
GPT2	0.12 ± 0.10	5.10 ± 2.06	0.40	0.56	0.67	0.55 ± 0.08	21.33 ± 5.21	0.40	0.60	0.71
FUDGE	0.30 ± 0.12	5.59 ± 0.60	0.39	0.55	0.65	0.57 ± 0.08	24.27 ± 3.53	0.40	0.60	0.70
MUCOLA	0.58 ± 0.23	33.09 ± 36.32	0.26	0.4	0.51	0.66 ± 0.08	85.74 ± 12.33	0.28	0.42	0.53
SVS-LANGEVIN	0.91 ± 0.12	14.26 ± 2.55	0.24	0.39	0.51	0.82 ± 0.06	26.76 ± 3.80	0.16	0.30	0.41
SVS	0.92 ± 0.05	13.9 ± 2.04	0.22	0.37	0.49	0.84 ± 0.06	32.73 ± 4.09	0.14	0.28	0.41
p-NCG	0.96 ± 0.03	6.82 ± 0.47	0.23	0.52	0.78	0.92 ± 0.05	39.03 ± 5.67	0.37	0.86	0.98
p-NCG + GwL	0.99 ± 0.02	5.17 ± 0.38	0.20	0.44	0.68	0.96 ± 0.04	23.61 ± 2.09	0.35	0.83	0.97

Table 1. Evaluation of different sampling methods on topic and sentiment controlled generation, using three criteria: Success at following the control target given by an external classifier (main metric), fluency (measured by perplexity), and diversity (measured by Distinct-n).

have similar performance, perhaps because the toy language model is too small to distinguish these samplers. Still, we can observe that all samplers except for MUCOLA are able to converge to the correct limiting distribution π_{toy}, albeit at different rates. Finally, we note that MUCOLA displays the systematic bias that we saw in Example 3.1, where we calculated its stationary distribution exactly through spectral decomposition. We see that MUCOLA's empirical distribution plateaus at a certain distance away from the true distribution.

5.2. Sampling from Language Models

Next, we test our methods on sampling from an unconstrained language model π, namely the GPT-2 checkpoint from the Huggingface library. We fix $N = 20$ and initialize the Markov chain with a random draw from Σ^N.

If P^t is an MCMC sampler's distribution at time step t, then the expected energy of its samples gives the cross-entropy $H(P^t, \pi)$ (in nats), plus a constant that depends only on π. Fig. 3 plots estimates of this expected energy for different samplers with $N = 20$—as well as for exact sampling from π, which gives the minimum possible value, achieved only when $P_t = \pi$. We observe that for the faithful samplers, p-NCG and GwL, P^t quickly converges to π. On the other hand, with the unfaithful MUCOLA sampler, P^t does not reach π, although its cross-entropy still decreases initially.

6. Conditional Sampling Experiments

We now try our methods on 3 controlled generation tasks in English. See App. G for full experimental setup details.

6.1. Tasks

Topic-Controlled Generation. Here our language model $p_{\text{LM}}(\boldsymbol{x})$ is a version of GPT-2-small that has been fine-tuned on the restaurant reviews in the E2E dataset (Novikova et al., 2017). We also use E2E's supervised annotations to train a stochastic classifier $p_{\text{CLS}}(t \mid \boldsymbol{x})$ that predicts the food type $t \in \{\text{Italian}, \text{Fast\_food}, \text{Japanese}, \ldots\}$ given review text $\boldsymbol{x}$.

	PPL(↓)	Distinct-1(↑)	Distinct-2(↑)	Distinct-3(↑)
π_{English}	57.42 ± 13.04	0.43	0.91	0.99
MUCOLA	95.38 ± 23.64	0.40	0.87	0.99
SVS-LANGEVIN	79.13 ± 19.08	0.44	0.92	1.00
SVS	77.16 ± 18.78	0.42	0.91	0.99
p-NCG	71.46 ± 17.41	0.39	0.85	0.99
p-NCG + GwL	55.06 ± 9.53	0.40	0.90	0.99

Table 2. Results on position-constrained generation using the filtered COLLIE dataset (Yao et al., 2024). The metrics are as in Table 1. Success for this task is always 1 (every sampler always preserves the specified tokens).

We then ask the model to generate a review of a specific food type t by sampling from $p(\boldsymbol{x} \mid t) \propto p_{\text{LM}}(\boldsymbol{x}) \cdot p_{\text{CLS}}(t \mid \boldsymbol{x})$.

Sentiment-Controlled Generation. Here we use GPT-2-large without fine-tuning as our p_{LM}. Similar to the topic control task, we train a sentiment classifier on the SST2 dataset of movie reviews (Socher et al., 2013) and ask the model to generate text with positive or negative sentiment.

Position-Constrained Generation. This is an text infilling task again using GPT-2-large. We use the setup from COLLIE (Yao et al., 2024), a challenging constrained generation benchmark that contains multiple types of constraints. We use the positional constraint subset of the dataset and filter for tokenizer differences (see App. G.1 for an example and further details). Here, the model is asked to generate a fixed-length sequence where 3 positions are constrained to specified tokens, which naturally yields an energy-based model. Each example is obtained by masking all but 3 tokens in a human-authored English sentence (from Project Gutenberg). The original unmasked sentence may be regarded as a draw from π_{English} (the conditional distribution of actual English), which should be similar to the target distribution π (the conditional distribution given by GPT-2-large), so we compare with that "sampler" in Table 2.

6.2. Baselines

We compare as applicable against the following baselines.

FUDGE. Introduced by Yang and Klein (2021), FUDGE samples tokens from the language model autoregressively, but weights the token probabilities at each position according to a classifier that determines whether the next token is likely to satisfy the constraint. In effect, by training classifiers to re-weight the per-step token probabilities under some global constraint, FUDGE is distilling a globally-normalized EBM into a locally normalized one, which Yang and Klein (2021) aptly referred to as "Future Discriminators".

Note that, since FUDGE requires a training dataset and COLLIE does not supply one, FUDGE is absent from Table 2.

MUCOLA. Introduced by Kumar et al. (2022), MUCOLA forms a Markov chain using the update equation in Eq. (9) and defines the energy function as

$$U(\boldsymbol{x}) = -\log p_{\text{LM}}(\boldsymbol{x}) - \beta \log p_{\text{CLS}}(t \mid \boldsymbol{x}) \qquad (20)$$

where β is a hyperparameter intended to balance the classifier energy and the language model.

SVS and SVS-LANGEVIN. Introduced by Amini et al. (2023), both methods define a piecewise continuous distribution based on the Voronoi cells generated from the word embeddings. SVS-LANGEVIN samples from this distribution using Langevin Dynamics, and SVS applies the appropriate form of HMC (Mohasel Afshar and Domke, 2015).

6.3. Evaluation

We sample multiple generations for each task (details in App. G) and evaluate them based on the following three downstream task metrics (when applicable):

1. **Success** is defined as the proportion of generations that were classified as having the desired topic or sentiment. To compare with previous papers, we evaluate this using a separately trained high-quality classifier.[16]
2. **Fluency** is measured by the perplexity under the language model.
3. **Distinct-n** is an indicator of *diversity*, which measures the type/token ratio of n-grams in a set of generated samples.

The results are in Tables 1 and 2. We also display generated samples for each sampler in Table 3 in App. H.

In tasks with control targets (i.e., topic- and sentiment-controlled generation), we can see that both p-NCG and its hybrid variant with GwL succeed in following the target at a much higher rate than baselines while maintaining a high level of fluency. Notably, the hybrid sampler p-NCG + GwL maintains a level of fluency comparable to the unconditional language model while adhering to the control target. In this respect, its sampling distribution resembles the true EBM distribution. In contrast, FUDGE obtains high fluency but often ignores the control target, while SVS and SVS-LANGEVIN sacrifice fluency in exchange for better compliance with the control.

In position-constrained generation, we first note that the high π_{English} perplexity is due to domain shift: the source sentences are from the Gutenberg subset of COLLIE whereas GPT-2(-large) is trained on WebText (Radford et al., 2019). The inherent diversity of constraints likely results in the higher diversity scores on Distinct-n. Of all samplers, our p-NCG + GwL produces the most fluent generations as it has the lowest perplexity. We note the overall higher perplexity compared to the sentiment control results. As both tasks use GPT-2-large as p_{LM}, this suggests that positional constraints are harder for language models to satisfy. Indeed, for our specific constraints, Yao et al. (2024) reported a near 0 constraint satisfaction rate from few-shot prompting GPT-4.

7. Conclusions and Future Work

In this work, we proposed two novel gradient-based samplers for generating text from energy-based models. We analyzed and compared against previous work that we illustrated and proved to be unfaithful samplers, meaning that their limiting distribution is different from the text distribution they want to sample from. We investigated the theoretical properties of our proposed samplers and then demonstrated with experiments that they have better performance in realistic tasks on text generation in terms of both controllability as well as fluency.

Our methods are not really specific to text: fundamentally they sample from a fixed-dimensional discrete space $\mathcal{X} \overset{\text{def}}{=} \mathcal{V}^N \subset \mathbb{R}^{Nd}$. Thus, they could also be used on the deep energy-based models of Ngiam et al. (2011), which (like Markov random fields) are joint distributions over any N random variables.

When applied to text, a limitation of our samplers is that N is fixed in advance, leading to a finite sample space Σ^N rather than Σ^*. In fact, the gradient-based proposal distribution only aims to replace individual words in their current positions. To sample variable-length sentences and to increase the mobility of the sampler, we could extend our proposal distribution to propose additional moves such as insertions and deletions (Miao et al., 2019), or even full rewrites of $\boldsymbol{w}$ using an prompted language model (Forristal et al., 2023).

Other ways to extend our algorithms exist. For example, while we manually tune the step size α for each model, we may adapt automatic tuning methods as in Hoffman and Gelman (2014) that preserve detailed balance. We could also use proposal merging algorithms in Horowitz (1991).

[16] Since the EBM has no direct knowledge of that classifier, this metric evaluates text quality, not sampler quality.

Impact Statement

This paper presents work whose goal is to advance the state of text generation, with the possibility of conditioning on controlled objectives. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgements

The experiments in this work were carried out at the Advanced Research Computing at Hopkins (ARCH) core facility, which is supported by the National Science Foundation (NSF) grant number OAC 1920103. Afra Amini is supported by the ETH AI Center doctoral fellowship. We thank Justin T. Chiu for helpful discussion and comments; Alex Lew for suggesting the COLLIE task; and anonymous reviewers for valuable feedback.

References

Afra Amini, Li Du, and Ryan Cotterell. 2023. Structured Voronoi sampling. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Filippo Ascolani, Gareth O. Roberts, and Giacomo Zanella. 2024. Scalability of Metropolis-within-Gibbs schemes for high-dimensional Bayesian models.

Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. 2018. Pyro: Deep universal probabilistic programming. *Journal of Machine Learning Research*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Mylène Bédard. 2017. Hierarchical models: Local proposal variances for RWM-within-Gibbs and MALA-within-Gibbs. *Computational Statistics & Data Analysis*, 109:231–246.

Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. 2017. Stan: A probabilistic programming language. *Journal of statistical software*, 76(1).

Apostolos Chalkis, Vissarion Fisikopoulos, Elias P. Tsigaridas, and Haris Zafeiropoulos. 2021. Geometric algorithms for sampling the flux space of metabolic networks. In *International Symposium on Computational Geometry (SoCG 2021)*, pages 21:1–21:16.

Ben Cousins and Santosh Vempala. 2016. A practical volume algorithm. *Mathematical Programming Computation*, 8(2):133–160.

Benjamin R. Cousins and Santosh S. Vempala. 2014. Gaussian cooling and $o^*(n^3)$ algorithms for volume and Gaussian volume. *SIAM J. Comput.*, 47:1237–1273.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. 2020. Residual energy-based models for text generation. In *International Conference on Learning Representations*.

Persi Diaconis, Susan Holmes, and Radford M. Neal. 2000. Analysis of a nonreversible Markov chain sampler. *The Annals of Applied Probability*, 10(3):726 – 752.

Li Du, Lucas Torroba Hennigen, Tiago Pimentel, Clara Meister, Jason Eisner, and Ryan Cotterell. 2023. A measure-theoretic characterization of tight language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9744–9770, Toronto, Canada. Association for Computational Linguistics.

Yilun Du and Igor Mordatch. 2019. Implicit generation and modeling with energy based models. In *Advances in Neural Information Processing Systems*, volume 32.

Simon Duane, A. D. Kennedy, Brian J. Pendleton, and Duncan Roweth. 1987. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222.

Rick Durrett. 2019. *Probability: Theory and Examples*, 5th edition. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press.

Martin E. Dyer and Alan M. Frieze. 1988. On the complexity of computing the volume of a polyhedron. *SIAM J. Comput.*, 17:967–974.

Ioannis Z. Emiris and Vissarion Fisikopoulos. 2013. Efficient random-walk methods for approximating polytope

volume. *Proceedings of the thirtieth annual symposium on Computational geometry*.

Ioannis Z. Emiris and Vissarion Fisikopoulos. 2018. Practical polytope volume approximation. *ACM Transactions on Mathematical Software (TOMS)*, 44:1 – 21.

Jarad Forristal, Fatemehsadat Mireshghallah, Greg Durrett, and Taylor Berg-Kirkpatrick. 2023. A block metropolis-hastings sampler for controllable energy-based text generation. In *Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 403–413.

Stuart Geman and Donald Geman. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: An interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, Vancouver, Canada. Association for Computational Linguistics.

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. 2022. Exposing the implicit energy networks behind masked language models via Metropolis–Hastings. In *International Conference on Learning Representations*.

Will Grathwohl, Kevin Swersky, Milad Hashemi, David Duvenaud, and Chris Maddison. 2021. Oops I took a gradient: Scalable sampling for discrete distributions. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3831–3841. PMLR.

Ulf Grenander and Michael I. Miller. 1994. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 56(4):549–603.

W. K. Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Bryan He, Christopher De Sa, Ioannis Mitliagkas, and Christopher Ré. 2016. Scan order in Gibbs sampling: Models in which it matters and bounds on how much. In *Advances in Neural Information Processing Systems*, volume 29.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8):1771–1800.

Matthew D. Hoffman and Andrew Gelman. 2014. The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(47):1593–1623.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, Melbourne, Australia. Association for Computational Linguistics.

Roger A. Horn and Charles R. Johnson. 2012. *Matrix Analysis*, 2nd edition. Cambridge University Press.

Alan M. Horowitz. 1991. A generalized guided Monte Carlo algorithm. *Physics Letters B*, 268:247–252.

A. D. Kennedy. 1990. *The Theory of Hybrid Stochastic Algorithms*, pages 209–223. Springer US, Boston, MA.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. GeDi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sachin Kumar, Biswajit Paria, and Yulia Tsvetkov. 2022. Constrained sampling from language models via Langevin dynamics in embedding spaces. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu Jie Huang. 2007. Energy-Based Models. In *Predicting Structured Data*. The MIT Press.

David A. Levin and Yuval Peres. 2017. *Markov Chains and Mixing Times*, 2nd edition. American Mathematical Soc.

Chu-Cheng Lin, Aaron Jaech, Xin Li, Matthew R. Gormley, and Jason Eisner. 2021. Limitations of autoregressive models and their alternatives. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5147–5173, Online. Association for Computational Linguistics.

David J. C. MacKay. 2003. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press. Available from `http://www.inference.phy.cam.ac.uk/mackay/itila/`.

James Martens and Ilya Sutskever. 2010. Parallelizable sampling of Markov random fields. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 517–524, Chia Laguna Resort, Sardinia, Italy. PMLR.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 1953. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. CGMH: Constrained sentence generation by Metropolis-Hastings sampling. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*.

David Mimno and Laure Thompson. 2017. The strange geometry of skip-gram with negative sampling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2873–2878, Copenhagen, Denmark. Association for Computational Linguistics.

Fatemehsadat Mireshghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. 2022. Mix and match: Learning-free controllable text generation using energy language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 401–415.

Hadi Mohasel Afshar and Justin Domke. 2015. Reflection, refraction, and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems*, volume 28.

Radford M. Neal. 1993. *Probabilistic Inference using Markov chain Monte Carlo methods*. Department of Computer Science, University of Toronto Toronto, ON, Canada.

Radford M. Neal. 2011. MCMC using Hamiltonian dynamics. In Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng, editors, *Handbook of Markov Chain Monte Carlo*, chapter 5. Chapman and Hall/CRC.

M.E.J. Newman and G.T. Barkema. 1999. *Monte Carlo Methods in Statistical Physics*. Oxford: Clarendon Press.

Jiquan Ngiam, Zhenghao Chen, Pang Wei Koh, and Andrew Ng. 2011. Learning deep energy models. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 1105–1112, New York, NY, USA. ACM.

Akihiko Nishimura, David B. Dunson, and Jianfeng Lu. 2020. Discontinuous Hamiltonian Monte Carlo for discrete parameters and discontinuous likelihoods. *Biometrika*, 107(2):365–380.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany. Association for Computational Linguistics.

Ari Pakman and Liam Paninski. 2013. Auxiliary-variable exact Hamiltonian Monte Carlo samplers for binary distributions. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Ari Pakman and Liam Paninski. 2014. Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians. *Journal of Computational and Graphical Statistics*, 23(2):518–542.

Du Phan, Neeraj Pradhan, and Martin Jankowiak. 2019. Composable effects for flexible and accelerated probabilistic programming in NumPyro. *arXiv preprint arXiv:1912.11554*.

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. 2022. COLD decoding: Energy-based constrained text generation with Langevin dynamics. In *Advances in Neural Information Processing Systems*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Benjamin Rhodes and Michael U. Gutmann. 2022. Enhanced gradient-based MCMC in discrete spaces. *Transactions on Machine Learning Research*.

Christian P. Robert and George Casella. 2004. *Monte Carlo Statistical Methods*. Springer New York, New York, NY.

Gareth O. Roberts and Jeffrey S. Rosenthal. 1998. Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60.

Gareth O. Roberts and Richard L. Tweedie. 1996. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341 – 363.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural*

Language Processing, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Jascha Sohl-Dickstein, Mayur Mudigonda, and Michael DeWeese. 2014. Hamiltonian Monte Carlo without detailed balance. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 719–726, Bejing, China. PMLR.

Yang Song and Stefano Ermon. 2020. Improved techniques for training score-based generative models. In *Advances in Neural Information Processing Systems*, volume 33, pages 12438–12448. Curran Associates, Inc.

Haoran Sun, Hanjun Dai, and Dale Schuurmans. 2022. Optimal scaling for locally balanced proposals in discrete spaces. In *Advances in Neural Information Processing Systems*.

X. T. Tong, M. Morzfeld, and Y. M. Marzouk. 2020. MALA-within-Gibbs samplers for high-dimensional distributions with sparse conditional structure. *SIAM Journal on Scientific Computing*, 42(3):A1765–A1788.

Max Welling and Yee Whye Teh. 2011. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, page 681–688, Madison, WI, USA. Omnipress.

Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, Online. Association for Computational Linguistics.

Shunyu Yao, Howard Chen, Austin W. Hanjie, Runzhe Yang, and Karthik R Narasimhan. 2024. COLLIE: Systematic construction of constrained text generation tasks. In *The Twelfth International Conference on Learning Representations*.

Giacomo Zanella. 2020. Informed proposals for local MCMC in discrete spaces. *Journal of the American Statistical Association*, 115(530):852–865.

Ruqi Zhang, Xingchao Liu, and Qiang Liu. 2022. A Langevin-like sampler for discrete distributions. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 26375–26396. PMLR.

Yichuan Zhang, Zoubin Ghahramani, Amos J. Storkey, and Charles Sutton. 2012. Continuous relaxations for discrete Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

A. Related Work

Controlled Generation. Since the introduction of large pretrained language models, controlled generation, the ability to enforce controls during the text generation process has become an important research direction (Keskar et al., 2019; Dathathri et al., 2020; Krause et al., 2021, *inter alia*). Earlier approaches in this direction include weighted decoding (Ghazvininejad et al., 2017; Holtzman et al., 2018; Yang and Klein, 2021), which adjusts the language model score of each sequence with a function that measures how well it adheres to its control objectives and then try to decode the high scoring sequences. More recently, several papers formulated energy-based models using pretrained language models (Deng et al., 2020; Goyal et al., 2022) to express the control objective (Kumar et al., 2022; Qin et al., 2022; Amini et al., 2023; Mireshghallah et al., 2022) and attempted to apply MCMC algorithms to sample from such sequence distribution. When the underlying pretrained language model is a masked language model (Mireshghallah et al., 2022), the masked distributions are highly effective as approximations to the true conditionals, and hence the Metropolis–Hastings corrected Gibbs-like scheme may work well without the need of gradient (Goyal et al., 2022). However, when the underlying is causal (Kumar et al., 2022; Qin et al., 2022; Amini et al., 2023), which is the subject of this paper, there is no obvious choice of proposal distributions as discussed in §3.1, and hence gradient information becomes valuable for deriving a proposal distribution without additional training.

Gradient-based Sampling Our work is also related to the line of research that makes use of gradient information to sample from complex distributions (Duane et al., 1987; Neal, 1993; Grenander and Miller, 1994). In Bayesian inference, gradient-based samplers (Neal, 2011; Hoffman and Gelman, 2014) are known to be highly effective when sampling from high-dimensional continuous distributions (Carpenter et al., 2017; Bingham et al., 2018; Phan et al., 2019). But it has been shown to be a difficult problem to adapt these algorithms in the discrete setting (Roberts and Tweedie, 1996; Roberts and Rosenthal, 1998), with previous approaches including continuous relaxation within the discrete spaces (Pakman and Paninski, 2013) using discontinuous Hamiltonian Monte Carlo (Pakman and Paninski, 2014; Mohasel Afshar and Domke, 2015; Nishimura et al., 2020), continuous relaxation via the "Gaussian Integral Trick" (Martens and Sutskever, 2010; Zhang et al., 2012). Specifically, the p-NCG proposed in our work is a generalization of NCG proposed in Rhodes and Gutmann (2022) and the D-MALA proposed in Zhang et al. (2022), with the difference being using the p-norm constraint instead of the standard ℓ_2 norm. The Gibbs-with-Langevin algorithm has its continuous analogue called MALA-within-Gibbs (Bédard, 2017; Tong et al., 2020) and is more generally an instance of within-Gibbs sampler (Robert and Casella, 2004, §10.3.3; Ascolani et al., 2024; *inter alia*). GwL is also loosely related to the Gibbs-with-Gradient method proposed in Grathwohl et al. (2021), which we found to have a near zero acceptance rate when applied to our setting. We note that a range of recently proposed gradient-based samplers (Grathwohl et al., 2021; Zhang et al., 2022; Rhodes and Gutmann, 2022) are all connected to the locally balanced proposal from (Zanella, 2020).

B. On High-Dimensional Integration in Embedding Spaces

B.1. The Problem of Continuous Relaxation and High-Dimensional Integration

A common strategy for continuous relaxation of discrete spaces is to map the discrete points into a continuous space and apply continuous gradient-based sampling algorithms (Pakman and Paninski, 2013; Amini et al., 2023). This strategy gives rise to the problem of converting samples from continuous algorithms into discrete ones. This problem is easier when the underlying discrete space is regularly shaped as in Ising model (Pakman and Paninski, 2013) where the projection function is as simple as the sign function $\mathrm{sgn}(\cdot)$. When the underlying discrete space is irregularly shaped such as the word embedding space, one can use the Euclidean projection to convert a continuous sample $\boldsymbol{y} \in \mathbb{R}^d$ into a discrete one $\boldsymbol{x} \in \mathcal{X}$, as in

$$\boldsymbol{x} = \mathrm{Proj}_{\mathcal{X}} \boldsymbol{y}. \tag{21}$$

This projection is used in both Amini et al. (2023) and Kumar et al. (2022) and it creates a number of problems.

SVS. In the case of SVS, Amini et al. (2023) realized that the projection created a piecewise continuous relaxation, with each continuous region corresponding to a Voronoi cell

$$V_i = \{\boldsymbol{y} \ : \ \|\boldsymbol{y} - \boldsymbol{x}_i\|_2 \leq \|\boldsymbol{y} - \boldsymbol{x}_{i'}\|_2, \forall\, i' \neq i\} \tag{22}$$

centering at a word embedding $\boldsymbol{x}_i$. Amini et al. (2023) then uses Gaussian augmentation within the Voronoi cells to apply gradient-based samplers. To ensure that the continuously relaxed measure matches original the discrete measure, the

underlying measure needs to be adjusted by the integral of the Gaussian truncated by the high-dimensional Voronoi polytope, otherwise known as the Gaussian volume of a polytope, defined as

$$\int_{V_i} \gamma^d(\boldsymbol{y}; \boldsymbol{x}, \sigma^2) \mathrm{d}\boldsymbol{y}. \tag{23}$$

where $\gamma^d(\,\cdot\,; \boldsymbol{x}, \sigma^2)$ denotes the d-dimensional Gaussian density centered at $\boldsymbol{x}$ with variance σ^2.

MuCoLa. By using the Euclidean projection operator in its update equation:

$$\boldsymbol{x}' = \mathrm{Proj}_{\mathcal{X}}\Big(\underbrace{\boldsymbol{x} - \frac{\alpha}{2}\nabla U(\boldsymbol{x})}_{\boldsymbol{\mu}_{\boldsymbol{x}}} + \sqrt{\alpha}\,\boldsymbol{\xi} \Big) \tag{9}$$

MuCoLa similarly identifies each Voronoi region in $\mathbb{R}^d$ with the word embedding at its center. As we demonstrated in §3.3, MuCoLa doesn't sample from its intended language distribution. An obvious idea is then to apply Metropolis-Hasting correction to MuCoLa, which requires one to compute $q_{\text{MuCoLa}}(\boldsymbol{x}_j \mid \boldsymbol{x}_i)$ in the Metropolis-Hasting acceptance probability Eq. (28). Observing that

$$\mathrm{Proj}_{\mathcal{X}}\left(\boldsymbol{x}_i - \frac{\alpha}{2}\nabla U(\boldsymbol{x}_i) + \sqrt{\alpha}\boldsymbol{\xi}\right) = \boldsymbol{x}_j \;\Leftrightarrow\; \boldsymbol{x}_i - \frac{\alpha}{2}\nabla U(\boldsymbol{x}_i) + \sqrt{\alpha}\boldsymbol{\xi} \in V_j, \tag{24}$$

we realize that computing $q_{\text{MuCoLa}}(\boldsymbol{x}_j \mid \boldsymbol{x}_i)$ is equivalent to computing the following integral

$$\int_{V_j} \gamma^d\left(\boldsymbol{y}; \boldsymbol{x}_i - \frac{\alpha}{2}\nabla U(\boldsymbol{x}_i), 1\right) \mathrm{d}\boldsymbol{y} \tag{25}$$

which is again the same high dimensional integral we encountered in SVS.

B.2. The Difficulty of High-Dimensional Integration

In general, computing the volume of an explicit polytope is #P-hard (Dyer and Frieze, 1988), which makes exact computation infeasible for dimensions as high as that of GPT-2 or BERT. Recent research on approximated high-dimensional integration shows great promise (Cousins and Vempala, 2014; Emiris and Fisikopoulos, 2013), and such algorithms (Cousins and Vempala, 2016; Emiris and Fisikopoulos, 2018) have improved to the extent that they can be employed in various applied sciences (Chalkis et al., 2021). Unfortunately, in our experimentation with these algorithms, we found that they can barely scale to dimensions beyond 100, not to mention the dimensions in GPT-2 or BERT, which are at the scale of 10^3. We, therefore, conclude that, at the current moment, the state of research in high-dimensional integration doesn't yet allow us to feasibly compute the relevant quantities so that SVS and MuCoLa can sample from the correct distribution.

C. Background: MCMC

C.1. Overview

Markov Chain Monte Carlo (MCMC; Metropolis et al. 1953) is based on the idea that to produce samples from a target distribution $\pi(x)$, one can design a transition kernel $p(x' \mid x)$ such that the resulting Markov chain converges to the target distribution. Intuitively, to guarantee that the target distribution is the limiting distribution of the Markov chain, one requires that the chain to be able to explore the entire state space and that the target distribution is invariant under the transition kernel.[17] The invariance condition is often algorithmically achieved by the *Metropolis-Hastings acceptance* procedure (App. C.3), which can adapt *any* Markov kernel into one that has the target distribution as a stationary distribution. Specifically, the Metropolis-Hastings procedure guarantees *reversibility* (or *detailed balance*), which is a stronger condition than invariance. Unless otherwise stated, all our proposed MCMC algorithms are corrected by Metropolis-Hastings acceptance. Finally, we often wish to design MCMC procedures that converge to the stationary distribution faster. This is measured by the *mixing time* (App. C.4), defined in Eq. (29).

[17] We provide an informal proof in App. C.2 of why these two criteria are sufficient.

C.2. Criteria for Convergence

Markov Chain Monte Carlo (MCMC; Metropolis et al. 1953) is based on the idea that to produce samples from a target distribution $\pi(x)$, one can design a transition kernel $p(x' \mid x)$ such that the resulting Markov chain has the target distribution as its limiting distribution. In finite discrete spaces, such as sampling sentences up to a fixed length, one designs the MCMC transition kernel to satisfy the following two criteria to guarantee convergence to then intended distribution $\pi(x)$ (such as the EBM of Eq. (2)):

(C1) *The chain is ergodic.* This means that, regardless of the starting state, the chain has a nonzero probability of being at every state after a sufficient number of steps. Ergodicity is equivalent to being irreducible and aperiodic.

(C2) *The target distribution is invariant under the transition kernel.* This means that, if the chain starts with the target distribution, it will stay in the target distribution, i.e.

$$\pi(x) = \sum_y p(x \mid y)\pi(y). \tag{26}$$

The reason that the above two criteria guarantee convergence to the target distribution is very simple. First of all, all finite state Markov chains have at least one stationary distribution. Adding the ergodicity requirement (C1) guarantees that the chain has a unique stationary distribution and the chain converges to that distribution, and (C2) ensures that the target distribution $\pi(x)$ *is* this unique stationary distribution. Therefore, (C1) and (C2) combined imply that the chain will always converge to the target distribution regardless of its starting state.

In practice, (C2) is often proved by establishing the detailed balance equation

$$\pi(x)p(x' \mid x) = \pi(x')p(x \mid x') \tag{27}$$

which implies that $\pi(x)$ is a stationary distribution of $p(\cdot \mid \cdot)$. When Eq. (27) holds for a given Markov chain $p(\cdot \mid \cdot)$, we also say that the chain is reversible with respect to distribution $\pi(\cdot)$ and $\pi(\cdot)$ is called a *reversing distribution* for $p(\cdot \mid \cdot)$.

Algorithmically, detailed balance (Eq. (27)) is often achieved by using the Metropolis–Hastings acceptance procedure (Metropolis et al., 1953; Hastings, 1970).

C.3. Metropolis–Hastings Acceptance

Metropolis–Hastings acceptance is a procedure to convert *any* Markov kernel $q(\cdot \mid \cdot)$ over $\mathcal{X}$, called a *proposal distribution*, into one that has the target distribution as its stationary. In each iteration, it draws a sample x' from $q(\cdot \mid x)$ and then *accepts* x' with the *acceptance probability*

$$\alpha(x' \mid x) = \min\left\{1, \frac{\pi(x')q(x \mid x')}{\pi(x)q(x' \mid x)}\right\}. \tag{28}$$

In the case x' is rejected, the chain remains at x. One easily checks that the chain derived from the acceptance procedure $p(x' \mid x) = \alpha(x' \mid x)q(x' \mid x)$ is a reversible chain with $\pi(\cdot)$ as its reversing distribution.

In this work, unless otherwise stated, all our algorithms are corrected with Metropolis–Hastings and hence we only need to specify the proposal distribution $q(\cdot \mid \cdot)$. However, it is important to point out that Metropolis–Hastings isn't always necessary. For example, by sampling from the true conditional, Gibbs sampling has a constant acceptance probability of 1, and hence the Metropolis–Hastings step can be omitted. One may alternatively design an irreversible Markov kernel that directly satisfies (C2) without satisfying Eq. (27) (see, e.g., Sohl-Dickstein et al., 2014; Diaconis et al., 2000).

C.4. Mixing Time

We wish to design MCMC algorithms that converge to the target distribution in a reasonable amount of time, and hence another important property of a given Markov chain is how fast it converges to the stationary distribution. This quantity is measured by the *mixing time*, t_{mix}. Denoting $P^t_{\boldsymbol{x}}$ as the t^{th} step distribution of a Markov chain started at state $\boldsymbol{x}$, the ε-mixing time is defined as

$$t_{\text{mix}}(\varepsilon) = \inf\left\{t : \sup_{\boldsymbol{x}\in\mathcal{X}} d_{\text{TV}}(P^t_{\boldsymbol{x}}, \pi) \le \varepsilon\right\} \tag{29}$$

where $d_{\text{TV}}(\cdot, \cdot)$ is the total variation distance[18] and π is the stationary distribution of the Markov chain. In words, $t_{\text{mix}}(\varepsilon)$ is the minimum number of steps necessary to achieve $\leq \varepsilon$ distance to the stationary distribution regardless of the starting state $\boldsymbol{x}$.

D. Derivation of p-NCG

We start with Eq. (10b)

$$q(\boldsymbol{x}' \mid \boldsymbol{x}) = \exp\left(-\frac{1}{2\alpha}\left\|\boldsymbol{x}' - \left(\boldsymbol{x} - \frac{\alpha}{2}\nabla U(\boldsymbol{x})\right)\right\|_2^2\right) \tag{30a}$$

$$= \exp\left(-\frac{1}{2\alpha}\left\|(\boldsymbol{x}' - \boldsymbol{x}) + \frac{\alpha}{2}\nabla U(\boldsymbol{x})\right\|_2^2\right) \tag{30b}$$

where

$$\frac{1}{2\alpha}\left\|(\boldsymbol{x}' - \boldsymbol{x}) + \frac{\alpha}{2}\nabla U(\boldsymbol{x})\right\|_2^2 \tag{31a}$$

$$= \frac{1}{2\alpha}\|\boldsymbol{x}' - \boldsymbol{x}\|_2^2 + 2 \cdot \frac{1}{2\cancel{\alpha}}\left\langle \boldsymbol{x}' - \boldsymbol{x}, \frac{\cancel{\alpha}}{2}\nabla U(\boldsymbol{x})\right\rangle + \frac{1}{2\alpha} \cdot \frac{\alpha^2}{4}\|\nabla U(\boldsymbol{x})\|_2^2 \tag{31b}$$

$$= \nabla U(\boldsymbol{x})^\top(\boldsymbol{x}' - \boldsymbol{x}) + \frac{1}{2\alpha}\|\boldsymbol{x}' - \boldsymbol{x}\|_2^2 + \frac{\alpha}{8}\|\nabla U(\boldsymbol{x})\|_2^2 \tag{31c}$$

Substituting Eq. (31c) into Eq. (30b), we get

$$q(\boldsymbol{x}' \mid \boldsymbol{x}) \propto \exp\left(-\nabla U(\boldsymbol{x})^\top(\boldsymbol{x}' - \boldsymbol{x}) - \frac{1}{2\alpha}\|\boldsymbol{x}' - \boldsymbol{x}\|_2^2 - \frac{\alpha}{8}\|\nabla U(\boldsymbol{x})\|_2^2\right) \tag{32}$$

Notice that the last term $\frac{\alpha}{8}\|\nabla U(\boldsymbol{x})\|_2^2$ only contains $\boldsymbol{x}$ and does not involve $\boldsymbol{x}'$, so it will cancel with the same term in the normalizing constant. This means that we can omit this term from the proposal distribution. Taking this into account, we get the alternate form of the proposal as given in Eq. (11):

$$q(\boldsymbol{x}' \mid \boldsymbol{x}) \propto \exp\left(-\nabla U(\boldsymbol{x})^\top(\boldsymbol{x}' - \boldsymbol{x}) - \frac{1}{2\alpha}\|\boldsymbol{x}' - \boldsymbol{x}\|_2^2\right). \tag{33}$$

E. Proof of Theorem 4.2

Theorem 4.2. *Let $\pi(\boldsymbol{x})$ be a discrete log-quadratic distribution as defined in Def. 4.1. For any $\alpha > 0$, there exists a unique distribution $\pi_\alpha(\boldsymbol{x})$ such that the Markov chain defined by q in Eq. (12) is reversible with respect to π_α and thus π_α is its stationary distribution. Further, $\pi_\alpha \to \pi$ weakly as $\alpha \to 0$.*

We adapt the proof strategy from the proof of Theorem 1 in Zanella (2020) and from Zhang et al. (2022).

Proof. To avoid confusion, we use $q_\alpha(\cdot \mid \boldsymbol{x})$ to denote the proposal in Eq. (12) with step size α, i.e.,

$$q_\alpha(\boldsymbol{x}' \mid \boldsymbol{x}) \propto \exp\left(-\frac{1}{2}\nabla U(\boldsymbol{x})^\top(\boldsymbol{x}' - \boldsymbol{x}) - \frac{1}{2\alpha}\|\boldsymbol{x}' - \boldsymbol{x}\|_p^p\right). \tag{34}$$

We first note that, for $\alpha > 0$, the proposal q_α is dense in the sense that $q_\alpha(\boldsymbol{x}' \mid \boldsymbol{x}) > 0$ for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$. This implies that the chain is irreducible and aperiodic, which guarantees that there must be a unique stationary distribution.

Let $\pi(\boldsymbol{x}) \propto \exp\left(\boldsymbol{x}^\top A\boldsymbol{x} + \boldsymbol{b}^\top\boldsymbol{x}\right)$ be a discrete log-quadratic distribution. In this case, the energy function is $U(\boldsymbol{x}) = -\boldsymbol{x}^\top A\boldsymbol{x} - \boldsymbol{b}^\top\boldsymbol{x}$. Since $U(\boldsymbol{x})$ is a quadratic function, the second-order Taylor expansion is exact, which means

$$U(\boldsymbol{x}') = U(\boldsymbol{x}) + \nabla U(\boldsymbol{x})^\top(\boldsymbol{x}' - \boldsymbol{x}) + \frac{1}{2}(\boldsymbol{x}' - \boldsymbol{x})^\top\nabla^2 U(\boldsymbol{x})(\boldsymbol{x}' - \boldsymbol{x}). \tag{35}$$

[18] Recall that the total variation distance is defined as $d_{\text{TV}}(\mu, \nu) \stackrel{\text{def}}{=} \sup_E |\mu(E) - \nu(E)|$.

Rearranging Eq. (35), we get

$$\nabla U(\boldsymbol{x})^\top(\boldsymbol{x}'-\boldsymbol{x}) = U(\boldsymbol{x}') - U(\boldsymbol{x}) - \frac{1}{2}(\boldsymbol{x}'-\boldsymbol{x})^\top\nabla^2 U(\boldsymbol{x})(\boldsymbol{x}'-\boldsymbol{x}) \tag{36}$$

which is equivalent to

$$\begin{aligned}\frac{1}{2}\nabla U(\boldsymbol{x})^\top(\boldsymbol{x}'-\boldsymbol{x}) &= \frac{1}{2}\left(U(\boldsymbol{x}') - U(\boldsymbol{x})\right) - \frac{1}{4}(\boldsymbol{x}'-\boldsymbol{x})^\top\nabla^2 U(\boldsymbol{x})(\boldsymbol{x}'-\boldsymbol{x}) &(37)\\ &= \frac{1}{2}\left(U(\boldsymbol{x}') - U(\boldsymbol{x})\right) + \frac{1}{2}(\boldsymbol{x}'-\boldsymbol{x})^\top A(\boldsymbol{x}'-\boldsymbol{x}). \qquad (\nabla^2 U(\boldsymbol{x}) = -2A) &(38)\end{aligned}$$

Using Eq. (38), we can rewrite the proposal Eq. (34) as

$$q_\alpha(\boldsymbol{x}' \mid \boldsymbol{x}) = \frac{1}{Z_\alpha(\boldsymbol{x})}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{x}') - U(\boldsymbol{x})\right) - \frac{1}{2}(\boldsymbol{x}'-\boldsymbol{x})^\top A(\boldsymbol{x}'-\boldsymbol{x}) - \frac{1}{2\alpha}\|\boldsymbol{x}'-\boldsymbol{x}\|_p^p\right) \tag{39}$$

where

$$Z_\alpha(\boldsymbol{x}) = \sum_{\boldsymbol{y}\in\mathcal{X}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y}) - U(\boldsymbol{x})\right) - \frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x}) - \frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right). \tag{40}$$

Now, we suppose π_α is a reversing distribution with respect to q_α and try to solve it. First, by the definition of reversibility,

$$\pi_\alpha(\boldsymbol{x})q_\alpha(\boldsymbol{x}' \mid \boldsymbol{x}) = \pi_\alpha(\boldsymbol{x}')q_\alpha(\boldsymbol{x} \mid \boldsymbol{x}') \tag{41}$$

which, after substituting in Eq. (39), expands to

$$\begin{aligned}&\frac{\pi_\alpha(\boldsymbol{x})}{Z_\alpha(\boldsymbol{x})}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{x}') - U(\boldsymbol{x})\right) - \frac{1}{2}(\boldsymbol{x}'-\boldsymbol{x})^\top A(\boldsymbol{x}'-\boldsymbol{x}) - \frac{1}{2\alpha}\|\boldsymbol{x}'-\boldsymbol{x}\|_p^p\right)\\ =&\frac{\pi_\alpha(\boldsymbol{x}')}{Z_\alpha(\boldsymbol{x}')}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{x}) - U(\boldsymbol{x}')\right) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^\top A(\boldsymbol{x}-\boldsymbol{x}') - \frac{1}{2\alpha}\|\boldsymbol{x}-\boldsymbol{x}'\|_p^p\right)\end{aligned} \tag{42}$$

and simplifies to

$$\begin{aligned}&\frac{\pi_\alpha(\boldsymbol{x})}{Z_\alpha(\boldsymbol{x})}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{x}') - U(\boldsymbol{x})\right)\right) = \frac{\pi_\alpha(\boldsymbol{x}')}{Z_\alpha(\boldsymbol{x}')}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{x}) - U(\boldsymbol{x}')\right)\right) &(43)\\ \Leftrightarrow\quad &\frac{\pi_\alpha(\boldsymbol{x})}{Z_\alpha(\boldsymbol{x})}\exp(U(\boldsymbol{x})) = \frac{\pi_\alpha(\boldsymbol{x}')}{Z_\alpha(\boldsymbol{x}')}\exp(U(\boldsymbol{x}')) &(44)\\ \Leftrightarrow\quad &\frac{\pi_\alpha(\boldsymbol{x})}{Z_\alpha(\boldsymbol{x})}\cdot\frac{Z}{\exp(-U(\boldsymbol{x}))} = \frac{\pi_\alpha(\boldsymbol{x}')}{Z_\alpha(\boldsymbol{x}')}\cdot\frac{Z}{\exp(-U(\boldsymbol{x}'))} \qquad (Z \overset{\text{def}}{=} \textstyle\sum_{\boldsymbol{x}\in\mathcal{X}}\exp(-U(\boldsymbol{x}))) &(45)\\ \Leftrightarrow\quad &\frac{\pi_\alpha(\boldsymbol{x})}{Z_\alpha(\boldsymbol{x})\pi(\boldsymbol{x})} = \frac{\pi_\alpha(\boldsymbol{x}')}{Z_\alpha(\boldsymbol{x}')\pi(\boldsymbol{x}')}. \qquad (\pi(\boldsymbol{x}) = \exp(-U(\boldsymbol{x}))/Z) &(46)\end{aligned}$$

Eq. (46) shows that $\frac{\pi_\alpha(\boldsymbol{x})}{Z_\alpha(\boldsymbol{x})\pi(\boldsymbol{x})} = c_\alpha$ for some constant c_α for all $\boldsymbol{x}\in\mathcal{X}$. Noting that $\sum_{\boldsymbol{x}\in\mathcal{X}}\pi_\alpha(\boldsymbol{x}) = 1$, we can solve for c_α to be

$$1 = \sum_{\boldsymbol{x}\in\mathcal{X}}\pi_\alpha(\boldsymbol{x}) = \sum_{\boldsymbol{x}\in\mathcal{X}}c_\alpha Z_\alpha(\boldsymbol{x})\pi(\boldsymbol{x}) = c_\alpha\sum_{\boldsymbol{x}\in\mathcal{X}}Z_\alpha(\boldsymbol{x})\pi(\boldsymbol{x}) \tag{47}$$

which yields

$$c_\alpha = \frac{1}{\sum_{\boldsymbol{x}\in\mathcal{X}}Z_\alpha(\boldsymbol{x})\pi(\boldsymbol{x})} \tag{48}$$

and hence the reversing measure π_α should be

$$\pi_\alpha(\boldsymbol{x}) = \frac{Z_\alpha(\boldsymbol{x})\pi(\boldsymbol{x})}{\sum_{\boldsymbol{y}\in\mathcal{X}}Z_\alpha(\boldsymbol{y})\pi(\boldsymbol{y})}. \tag{49}$$

One can quickly verify that π_α as defined in Eq. (49) indeed satisfies the detailed balance equation in Eq. (41) and hence is indeed a reversing measure for q_α. We can now conclude that q_α produces a reversible chain and that π_α is its unique stationary (and simultaneously reversing) measure.[19]

Finally, to show the weak convergence, we observe that

$$\lim_{\alpha\to 0} \exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right)=\begin{cases}0 & \boldsymbol{y}\neq\boldsymbol{x}\\ 1 & \boldsymbol{y}=\boldsymbol{x}\end{cases}=\delta_{\boldsymbol{x}}(\boldsymbol{y}) \tag{50}$$

where $\delta_{\boldsymbol{x}}(\cdot)$ is the Dirac delta centered at $\boldsymbol{x}$. This means that

$$\lim_{\alpha\to 0} Z_\alpha(\boldsymbol{x}) \tag{51}$$

$$=\lim_{\alpha\to 0}\sum_{\boldsymbol{y}\in\mathcal{X}} \exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right) \tag{52}$$

$$=\sum_{\boldsymbol{y}\in\mathcal{X}}\delta_{\boldsymbol{x}}(\boldsymbol{y}) \qquad \text{(by Eq. (50))} \tag{53}$$

$$=1. \tag{54}$$

Hence

$$\lim_{\alpha\to 0}\pi_\alpha(\boldsymbol{x})=\lim_{\alpha\to 0}\frac{Z_\alpha(\boldsymbol{x})\pi(\boldsymbol{x})}{\sum_{\boldsymbol{y}\in\mathcal{X}} Z_\alpha(\boldsymbol{y})\pi(\boldsymbol{y})}=\frac{\pi(\boldsymbol{x})}{\sum_{\boldsymbol{y}\in\mathcal{X}}\pi(\boldsymbol{y})}=\pi(\boldsymbol{x}) \tag{55}$$

which shows that π_α converges to π pointwise. It is a well-known result that, in the case of discrete distributions, pointwise convergence implies weak convergence.[20] Hence, $\pi_\alpha\to\pi$ weakly as $\alpha\to 0$. □

F. Proof of Theorem 4.3

We state the Geršgorin disc theorem here for reference.

Theorem F.1 (Geršgorin disc theorem; Theorem 6.1.1 in Horn and Johnson, 2012)**.** *Given a matrix P and denote its non-diagonal sum as $R_i=\sum_{j\neq i}|P_{ij}|$. Define the Geršgorin discs as*

$$D(a_{ii},R_i)=\{z\in\mathbb{C}:|z-a_{ii}|\leq R_i\}. \tag{56}$$

Then, all eigenvalues of P are in the union of the Geršgorin discs.

Theorem 4.3. *Let $\pi(\boldsymbol{x})$ be a discrete log-quadratic distribution as defined in Def. 4.1. There exist constants $c_1,c_2,Z>0$ that depends only on $\pi(\boldsymbol{x})$ such that the mixing time of q in Eq. (12) satisfies*

$$t_{\text{mix}}(\varepsilon)\geq\left(\frac{c_1}{Z}\exp\left(\frac{c_2}{2\alpha}\right)-1\right)\log\left(\frac{1}{2\varepsilon}\right) \tag{15}$$

Proof. Let $\pi(\boldsymbol{x})\propto\exp\left(\boldsymbol{x}^\top A\boldsymbol{x}+\boldsymbol{b}^\top\boldsymbol{x}\right)$ be a discrete log-quadratic distribution. Here, we let the energy function be $U(\boldsymbol{x})=-\boldsymbol{x}^\top A\boldsymbol{x}-\boldsymbol{b}^\top\boldsymbol{x}+\text{const}$. We additionally assume, without loss of generality, that $U(\boldsymbol{x})\leq 0$ for all $\boldsymbol{x}\in\mathcal{X}$, since we can subtract a constant from the energy function of each state without altering the distribution.

We recall from the proof of Theorem 4.2 that the proposal can be rewritten as

$$q_\alpha(\boldsymbol{x}'\mid\boldsymbol{x})=\frac{1}{Z_\alpha(\boldsymbol{x})}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{x}')-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{x}'-\boldsymbol{x})^\top A(\boldsymbol{x}'-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{x}'-\boldsymbol{x}\|_p^p\right) \tag{39}$$

where

$$Z_\alpha(\boldsymbol{x})=\sum_{\boldsymbol{y}\in\mathcal{X}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right). \tag{40}$$

[19] One may notice at Eq. (43) that setting $\pi_\alpha(\boldsymbol{x})\propto\exp(-U(\boldsymbol{x}))/Z_\alpha(\boldsymbol{x})$ will symmetrize both sides of the equation, resulting in detailed balance. This observation can avoid the last bit of calculation.

[20] See, for example, Exercise 3.2.11 in Durrett (2019).

To apply the Geršgorin disc theorem, we first need to bound the non-diagonal mass in the transition matrix. The non-diagonal mass, i.e., the probability of non-self-transition, is

$$\sum_{\boldsymbol{y}\neq\boldsymbol{x}} q_\alpha(\boldsymbol{y}\mid\boldsymbol{x}) \tag{57}$$

$$=\frac{\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right)}{\sum_{\boldsymbol{y}\in\mathcal{X}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right)} \tag{58}$$

$$=\frac{\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right)}{1+\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right)} \tag{59}$$

$$\leq\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right) \tag{60}$$

Without loss of generality, we can assume that A is symmetric because we can substitute A with its symmetric part $\frac{1}{2}(A^\top+A)$ without changing any quantity of interest. Then we can apply the Rayleigh-Ritz inequality,which states that, for any $\boldsymbol{v}\neq 0$,

$$\frac{\boldsymbol{v}^\top A\boldsymbol{v}}{\boldsymbol{v}^\top\boldsymbol{v}}\geq\lambda_{\min}(A). \tag{61}$$

We further define the useful quantity for $q\geq 1$,

$$d_q\stackrel{\text{def}}{=}\inf_{\boldsymbol{x}\neq\boldsymbol{x}'\in\mathcal{X}}\|\boldsymbol{x}-\boldsymbol{x}'\|_q^q. \tag{62}$$

Continuing from Eq. (60),

$$\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{x})^\top A(\boldsymbol{y}-\boldsymbol{x})-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right) \tag{63}$$

$$\leq\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}\lambda_{\min}(A)\|\boldsymbol{y}-\boldsymbol{x}\|_2^2-\frac{1}{2\alpha}\|\boldsymbol{y}-\boldsymbol{x}\|_p^p\right) \quad \text{(Rayleigh-Ritz, Eq. (61))} \tag{64}$$

$$\leq\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}\left(U(\boldsymbol{y})-U(\boldsymbol{x})\right)-\frac{1}{2}\lambda_{\min}(A)d_2-\frac{1}{2\alpha}d_p\right) \quad \text{(definition of } d_q\text{, Eq. (62))} \tag{65}$$

$$=\exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2-\frac{1}{2\alpha}d_p\right)\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}U(\boldsymbol{y})+\frac{1}{2}U(\boldsymbol{x})\right) \tag{66}$$

$$\leq\exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2-\frac{1}{2\alpha}d_p\right)\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-\frac{1}{2}U(\boldsymbol{y})\right) \quad \text{(assumption that } U(\boldsymbol{x})\leq 0\text{)} \tag{67}$$

$$\leq\exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2-\frac{1}{2\alpha}d_p\right)\sum_{\boldsymbol{y}\neq\boldsymbol{x}}\exp\left(-U(\boldsymbol{y})\right) \quad \text{(assumption that } U(\boldsymbol{y})\leq 0\text{)} \tag{68}$$

$$\leq\exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2-\frac{1}{2\alpha}d_p\right)\sum_{\boldsymbol{y}\in\mathcal{X}}\exp\left(-U(\boldsymbol{y})\right) \tag{69}$$

$$=Z\exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2-\frac{1}{2\alpha}d_p\right). \quad \left(Z\stackrel{\text{def}}{=}\textstyle\sum_{\boldsymbol{x}\in\mathcal{X}}\exp(-U(\boldsymbol{x}))\right) \tag{70}$$

Combining Eq. (60) and Eq. (70), we obtain a bound for the non-self-transition probability

$$\sum_{\boldsymbol{y}\neq\boldsymbol{x}} q_\alpha(\boldsymbol{y}\mid\boldsymbol{x})\leq Z\exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2-\frac{1}{2\alpha}d_p\right). \tag{71}$$

We have established in Theorem 4.2 that the Markov chain defined by q_α is reversible. It is a well-known fact that the transition matrix of a reversible Markov chain has only real eigenvalues, and hence, the Geršgorin disc theorem (Theorem F.1)

in this specific case implies that an eigenvalue λ of the transition matrix of q_α satisfies

$$|\lambda - q_\alpha(\boldsymbol{x} \mid \boldsymbol{x})| \leq \sum_{\boldsymbol{y} \neq \boldsymbol{x}} q_\alpha(\boldsymbol{y} \mid \boldsymbol{x}) \leq Z \exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2 - \frac{1}{2\alpha}d_p\right) \tag{72}$$

for at least one of $\boldsymbol{x} \in \mathcal{X}$. In particular, λ_2, the 2nd largest eigenvalue of the transition matrix of q_α, satisfies, for at least one $\boldsymbol{x} \in \mathcal{X}$,

$$|\lambda_2 - q_\alpha(\boldsymbol{x} \mid \boldsymbol{x})| \leq Z \exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2 - \frac{1}{2\alpha}d_p\right). \tag{73}$$

In a reversible Markov chain, the *spectral gap* is defined as $\gamma = 1 - \lambda_2$ (Levin and Peres, 2017, §12.2). Using Eq. (71) and Eq. (73), we can bound the spectral gap with

$$\begin{aligned}
1 - \lambda_2 &= |1 - \lambda_2| && (1 = \lambda_1 \geq \lambda_2 \text{ in a reversible transition matrix}) && (74)\\
&\leq |1 - q_\alpha(\boldsymbol{x} \mid \boldsymbol{x})| + |q_\alpha(\boldsymbol{x} \mid \boldsymbol{x}) - \lambda_2| && (\text{triangle ineq.}) && (75)\\
&\leq |1 - q_\alpha(\boldsymbol{x} \mid \boldsymbol{x})| + Z \exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2 - \frac{1}{2\alpha}d_p\right) && (\text{Eq. (73)}) && (76)\\
&= \sum_{\boldsymbol{y} \neq \boldsymbol{x}} q_\alpha(\boldsymbol{y} \mid \boldsymbol{x}) + Z \exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2 - \frac{1}{2\alpha}d_p\right) && (q_\alpha(\cdot \mid \boldsymbol{x}) \text{ is a distribution}) && (77)\\
&\leq 2 \cdot Z \exp\left(-\frac{1}{2}\lambda_{\min}(A)d_2 - \frac{1}{2\alpha}d_p\right). && (\text{Eq. (71)}) && (78)
\end{aligned}$$

Finally, the mixing time and the spectral gap are closely related by the following well-known relationship.

Theorem F.2 (Theorem 12.4 and 12.5 in Levin and Peres, 2017)**.** *In a reversible, irreducible Markov chain, the spectral gap γ and the mixing time $t_{\text{mix}}(\varepsilon)$ are related by*

$$\left(\frac{1}{\gamma} - 1\right)\log\left(\frac{1}{2\varepsilon}\right) \leq t_{\text{mix}}(\varepsilon) \leq \frac{1}{\gamma}\log\left(\frac{1}{\varepsilon\pi_{\min}}\right) \tag{79}$$

where $\pi_{\min} = \min_{x \in \mathcal{X}} \pi(x)$.

Using the left inequality in Theorem F.2, we can conclude that

$$\begin{aligned}
t_{\text{mix}}(\varepsilon) &\geq \left(\frac{1}{\gamma} - 1\right)\log\left(\frac{1}{2\varepsilon}\right) && (80)\\
&= \left(\frac{1}{1 - \lambda_2} - 1\right)\log\left(\frac{1}{2\varepsilon}\right) && (81)\\
&\geq \left[\frac{1}{2 \cdot Z}\exp\left(\frac{1}{2}\lambda_{\min}(A)d_2 + \frac{1}{2\alpha}d_p\right) - 1\right]\log\left(\frac{1}{2\varepsilon}\right) && (82)\\
&= \left[\frac{\exp(\lambda_{\min}(A)d_2/2)}{2 \cdot Z}\exp\left(\frac{d_p}{2\alpha}\right) - 1\right]\log\left(\frac{1}{2\varepsilon}\right) && (83)
\end{aligned}$$

Setting $c_1 = \frac{1}{2}\exp(\lambda_{\min}(A)d_2/2) > 0$ and $c_2 = d_p > 0$, we obtain the desired bound

$$t_{\text{mix}}(\varepsilon) \geq \left(\frac{c_1}{Z}\exp\left(\frac{c_2}{2\alpha}\right) - 1\right)\log\left(\frac{1}{2\varepsilon}\right). \tag{84}$$

□

G. Experimental Setup

G.1. Data Setup

- For topic controlled task, for each method, we generate 20 samples of 15 tokens for each control target, resulting in a total 140 samples (from a total of 7 control targets).

- For sentiment controlled task, for each method, we generate 60 samples of 15 tokens for each sentiment, resulting in a total of 120 samples.
- For position constrained task, we use the arbitrary position constraint examples from Gutenberg sourced subset of the COLLIE dataset. In this subset, the model is tasked to generate a fixed-length sequence where 3 positions have specified tokens, e.g., the 4th, 7th and 10th tokens must be "shown", "could", and "far", respectively. The constraints are sourced from texts in human written materials, e.g., the above constraint is sourced from the sentence "But she had shown her that one could go too far.".
  To account for tokenization differences between GPT-2 and GPT-4, we further filter the examples such that the target words exist in GPT-2's tokenizer. This results in a total of 43 distinct constraint examples. We then use each method to generate 4 samples for each constraint, resulting in a total of 172 samples per method.

G.2. Hyperparameters

In all our experiments, we found that in GwL, random scan in general performs better than systematic scan. Therefore, all results reported for GwL uses random scan.

Choice of p. In our early experiments, we experimented with different values of p over a grid between 1.0 and 2.0 at intervals of 0.1 on unconditional sampling from GPT-2-large, and found that lower p values are better. The performance differences between $p \in [1.0, 1.2]$ are small, and hence for efficiency reasons, we choose to use $p = 1$, which amounts to a absolute value operation. All results are reported with $p = 1$.

Sampler configurations. For topic controlled experiments, which uses the smallest model, we run all samplers for 4,000 steps. For sentiment and position controlled experiments, we run all samplers for 10,000 steps.

All step sizes are tuned with grid search with a grid resolution of 0.1. For the Toy Example, the grid search objective is average energy. For the controlled generation tasks, the grid search objective is success rate.

- **Toy Example.** In the toy example, the inverse temperature $\beta = 0.42$ and the sequence length (i.e., the number of spins in the Ising model) is $N = 5$. The underlying Ising topology is a linear chain with the ends connected. The step size for MUCOLA is 1.5, the trajectory length of SVS is 2π, and the step size of p-NCG and GwL are both 1.0
- **Sampling from Language Models.** The step size for MUCOLA is 0.15, and the step size of both p-NCG and GwL is 4.0. Each chain is ran on 100 random seeds to estimate the error bars.
- **Topic and Sentiment Controlled Generation.** For all samplers, the energy weight used is $\beta = 25.0$. The step size used for MUCOLA is 1.0. For SVS and SVS-LANGEVIN, the step size is 1.5. Finally, for p-NCG and GwL, the step size is $\alpha = 1.0$ in topic controlled experiments and $\alpha = 3.0$ in sentiment controlled experiments.
- **Position controlled Generation.** The step size used for MUCOLA is 1.0. For SVS and SVS-LANGEVIN, the energy weight is $\beta = 1.5$ and the step size is 1.5. Finally, for p-NCG and GwL, the step size is $\alpha = 1.0$ and the energy is $\beta = 1.25$.

Classifiers. We train two classifiers independently, called an internal classifier and an external classifier. The internal classifier is used as the energy function during generation, and the external classifier is used to determine whether the generated text follows the control objective correctly.

The *internal classifier* is a probing classifier on top of frozen GPT-2 embeddings. The probing classifier is a 3-layered BILSTM model with 0.5 dropout. The classifier achieves a 0.84 F1 score on the test set. We then train an evaluator classifier to evaluate the success rates of the controlled generation algorithms.

The *external classifier* for topic controlled generation is a fine-tuned ROBERTA model that achieves 0.90 f1-score on the test set. For sentiment controlled generation, we use an off-the-shelf finetuned Transformer model, `distilbert-base-uncased-finetuned-sst-2-english`, from the HuggingFace library.

H. Controlled Generation Samples

We present controlled generation text samples in Table 3.

	Chinese
FUDGE	In the city centre near Yippee Noodle Bar Chinese, is Alimentum. It has moderate prices and
MUCOLA	and has a 1 out of 5. It has food and high customer rating. The Rice Boat is
SVS-LANGEVIN	It serves Chinese food with a low customer rating. The fast food and restaurant The Golden Curry is a
SVS	It has a low customer rating and a price. The highly rated Chinese restaurant The Phoenix has a high
p-NCG + GwL	The Golden Curry is a Chinese food restaurant with a 5 out 5 rating and is not family-friendly
	English
FUDGE	It has an average customer Rating. Bibimbap House has English food in the riverside area near
MUCOLA	and has a low customer rating. The Golden Curry is a children friendly, serving English food, with
SVS-LANGEVIN	It has low rating and is located near the to the city centre. The Phoenix is a English food
SVS	Alimentum in the city centre near the a moderate price range. It serves English food, is
p-NCG + GwL	Midsummer House serves English food with a moderate price range and a high customer rating. It is
	Fast food
FUDGE	A fast food, coffee shop, Strada has a low customer rating, has a price range of over £30. It is
MUCOLA	and is family friendly and serves fast food. The Wrestlers is a fast food coffee shop in the
SVS-LANGEVIN	It is located near the riverside, is a cheap family friendly fast food restaurant, and is called
SVS	It is located near the river. The Mill is a cheap, fast food and coffee shop near the
p-NCG + GwL	Alimentum is a high-priced, child friendly, average rated fast food restaurant that is in
	French
FUDGE	It has a low-priced Inn French food. It is near Café Rouge.The Alimentum is a kid friendly fast food
MUCOLA	The French restaurant The Waterman is located in the city centre. The price range is less than
SVS-LANGEVIN	It is a restaurant located in the riverside, the restaurant, offers French food with a price
SVS	It is a family restaurant that serves French food with a price range and has a low customer rating.
p-NCG + GwL	The Waterman, located in city centre, has average French food, is inexpensive and is not family
	Indian
FUDGE	The Phoenix Indian restaurant has moderate prices with a 3 out of 5 rating. Located on the
MUCOLA	It is in the city and has a low customer rating. The Waterman is a low priced
SVS-LANGEVIN	It is not child friendly and it is near the river. It serves Indian food and a customer rating
SVS	It is located in the city centre near The Portland Arms Indian food and has a low customer rating.
p-NCG + GwL	The Phoenix is in the city centre that provides Indian food in the cheap price range. Its customer rating
	Italian
FUDGE	It has family Italian food and has a low a moderate price range. The Rice Boat has an average
MUCOLA	is a high priced Italian food restaurant with a customer rating of average. The Phoenix is a high
SVS-LANGEVIN	It is located in the city centre, it is not family friendly and is a coffee shop serving Italian
SVS	It is located in the the city centre near The Portland Arms.The Eagle is an Italian restaurant.
p-NCG + GwL	The Eagle Italian food coffee shop, is a family friendly riverside restaurant with a low customer rating.
	Japanese
FUDGE	Japanese food. Its customer rating is 3 out of 5.The Phoenix is Japanese in the city centre
MUCOLA	for Japanese food is located in the city centre. It has a low customer rating. The Golden
SVS-LANGEVIN	It is located in the riverside. It is a Japanese food. It is a pub restaurant
SVS	It is located in the riverside. It is a low rated Japanese restaurant, and coffee shop.
p-NCG + GwL	It also serves Japanese food. It is located in the city centre and has a high price range.

Table 3. Examples of sampled sentences from different control food targets.