On Divergence Measures for Training GFlowNets

Tiago da Silva **Eliezer de Souza da Silva** **Diego Mesquita**
{tiago.henrique, eliezer.silva, diego.mesquita}@fgv.br
School of Applied Mathematics
Getulio Vargas Foundation
Rio de Janeiro, Brazil

Abstract

Generative Flow Networks (GFlowNets) are amortized samplers of unnormalized distributions over compositional objects with applications to causal discovery, NLP, and drug design. Recently, it was shown that GFlowNets can be framed as a hierarchical variational inference (HVI) method for discrete distributions. Despite this equivalence, attempts to train GFlowNets using traditional divergence measures as learning objectives were unsuccessful. Instead, current approaches for training these models rely on minimizing the log-squared difference between a proposal (forward policy) and a target (backward policy) distribution. In this work, we first formally extend the relationship between GFlowNets and HVI to distributions on arbitrary measurable topological spaces. Then, we empirically show that the ineffectiveness of divergence-based learning of GFlowNets is due to the large gradient variance of the corresponding stochastic objectives. To address this issue, we devise a collection of provably variance-reducing control variates for gradient estimation based on the REINFORCE leave-one-out estimator. Our experimental results suggest that the resulting algorithms often accelerate training convergence when compared against previous approaches. All in all, our work contributes by narrowing the gap between GFlowNet training and HVI, paving the way for algorithmic advancements inspired by the divergence minimization viewpoint.

1 Introduction

The approximation of intractable distributions is one of the central issues in machine learning and modern statistics [7, 35]. In reinforcement learning (RL), a recurring goal is to find a diverse set of high-valued state–action trajectories according to a reward function. This problem may be cast as sampling trajectories proportionally to the reward, which is generally an intractable distribution over the environment [3, 9, 38, 50]. Similarly, practical Bayesian inference and probabilistic models computations involve assessing intractable posterior distributions [36, 78, 102]. In the variational inference (VI) approach, circumventing this intractability involves searching for a tractable approximation to the target distribution within a family of parametric models. Conventionally, the problem reduces to minimizing a divergence measure, such as Kullback-Leibler (KL) divergence [7, 36, 92] or Renyi-α divergence [51, 70], between the variational approximation and the target.

In particular, Generative Flow Networks (GFlowNets) [3, 4, 48] are a recently proposed family of variational approximations well-suited for distribution over compositional objects (e.g., graphs and texts). GFlowNets have found empirical success within various applications from causal discovery [15, 16], NLP [30], and chemical and biological modeling [3, 32]. In a nutshell, a GFlowNet learns an iterative generative process (IGP) [26] over an extension of the target's support, which, for sufficiently expressive parameterizations of transition kernels, yields independent and correctly distributed samples [3, 48]. Remarkably, training GFlowNets typically consists of minimizing the log-squared difference between a proposal and target distributions over the extended space via SGD [4, 55], contrasting with divergence-minimizing algorithms commonly used in VI [7, 72].

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-2416

Indeed, Malkin et al. [56] suggests that trajectory balance (TB) loss training for GFlowNets leads to better approximations of the target distribution than directly minimizing the reverse and forward KL divergence, particularly in setups with sparser rewards. Nevertheless, as we highlight in Section 3, these results are a potential consequence of biases and high variance in gradient estimates for the divergence's estimates, which can be overlooked in the evaluation protocol reliant upon sparse target distributions. Therefore, in Section 5, we present a comprehensive empirical investigation of the minimization of well-known f-divergence measures (including reverse and forward KL), showing it is an effective procedure that often accelerates the training convergence of GFlowNets relative to alternatives. To achieve these results, we develop in Section 4 a collection of control variates (CVs) [63, 71] to reduce the variance without introducing bias on the estimated gradients, improving the efficiency of the optimization algorithms [77, 85]. In summary, our *main contributions* are:

1. We evaluate the performance of forward and reverse KL- [47], Renyi-α [74] and Tsallis-α [91] divergences as learning objectives for GFlowNets through an extensive empirical campaign and highlight that they frequently outperform traditionally employed loss functions.
2. We design control variates for the gradients of GFlowNets' divergence-based objectives. Therefore, it is possible to perform efficient evaluations of the optimization objectives using automatic differentiation frameworks [69], and the resulting experiments showcase the significant reduction in the variance of the corresponding estimators.
3. We developed a theoretical connection between GFlowNets and VI beyond the setup of finitely supported measures [56, 112], establishing results for arbitrary topological spaces.

2 Revisiting the relationship between GFlowNets and VI

Initially, we review Lahlou et al. [48]'s work on GFlowNets for distributions on topological spaces, a perspective applied consequentially to obtain the equivalence between GFlowNets training and VI divergence minimization in a more generic setting. Finally, we describe standard variance reduction techniques for solving stochastic optimization problems.

Notations. Let $(\mathcal{S}, \mathcal{T})$ be a topological space with topology $\mathcal{T}$ and Σ be the corresponding Borel σ-algebra. Also, let $\nu\colon \Sigma \to \mathbb{R}_+$ be a measure over Σ and $\kappa_f, \kappa_b\colon \mathcal{S} \times \Sigma \to \mathbb{R}_+$ be transition kernels over $\mathcal{S}$. For each $(B_1, B_2) \in \Sigma \times \Sigma$, we denote by $\nu \otimes \kappa(B_1, B_2) \coloneqq \int_{B_1} \nu(\mathrm{d}s) k(s, B_2)$. Likewise, we recursively define the *product kernel* as $\kappa^{\otimes 0}(s, \cdot) = \kappa(s, \cdot)$ and, for $n \geq 1$, $\kappa^{\otimes n}(s, \cdot) = \kappa^{\otimes n-1}(s, \cdot) \otimes \kappa$ for a transition kernel κ and $s \in \mathcal{S}$. Note, in particular, that $\kappa^{\otimes n}$ is a function from $\mathcal{S} \times \Sigma^{\otimes n+1}$ to $\mathbb{R}_+$, with $\Sigma^{\otimes n+1}$ representing the product σ-algebra of Σ [1, 96]. Moreover, if μ is an absolutely continuous measure relatively to ν, denoted $\mu \ll \nu$, we write $\mathrm{d}\mu/\mathrm{d}\nu$ for the corresponding density (Radom-Nikodym derivative) [1]. Furthermore, we denote by $\mathcal{P}(A) = \{S\colon S \subseteq A\}$ the power-set of a set $A \subset \mathcal{S}$ and by $[d] = \{1, \ldots, d\}$ the first d positive integers.

GFlowNets. A GFlowNet is, in its most general form, built upon the concept of a *measurable pointed directed acyclic graph* (DAG) [48], which we define next. Intuitively, it extends the notion of a *flow network* to arbitrary measurable topological spaces, replacing the directed graph with a transition kernel specifying how the underlying states are connected.

Definition 1 (Measurable pointed DAG [48])**.** Let $(\bar{\mathcal{S}}, \mathcal{T}, \Sigma)$ be a measurable topological space endowed with a reference measure ν and forward κ_f and backward κ_b kernels. Also, let $s_o \in \bar{\mathcal{S}}$ and $s_f \in \bar{\mathcal{S}}$ be distinguished elements in $\bar{\mathcal{S}}$, respectively called *initial* and *final* states, and $\mathcal{S} = \bar{\mathcal{S}} \setminus \{s_f\}$. We assume $\{s_f\}$ is open. A *measurable pointed DAG* is then a tuple $(\mathcal{S}, \mathcal{T}, \Sigma, \kappa_f, \kappa_b, \nu)$ satisfying:

1. **(Terminality)** If $\kappa_f(s, \{s_f\}) > 0$, then $\kappa_f(s, \{s_f\}) = 1\ \forall s \in \bar{\mathcal{S}}$. Also, $\kappa_f(s_f, \cdot) = \delta_{s_f}$.
2. **(Reachability)** For all $B \in \Sigma$, $\exists\, n \in \mathbb{N}$ s.t. $\kappa_f^{\otimes n}(s_o, B) > 0$, i.e., B is reachable from s_o.
3. **(Consistency)** For every $(B_1, B_2) \in \Sigma \times \Sigma$ such that $(B_1, B_2) \notin \{(s_o, s_o), (s_f, s_f)\}$, $\nu \otimes \kappa_f(B_1, B_2) = \nu \otimes \kappa_b(B_2, B_1)$. Moreover, $\kappa_b(s_o, B) = 0$ for every $B \in \Sigma$.
4. **(Continuity)** $s \mapsto \kappa_f(s, B)$ is continuous for $B \in \Sigma$.
5. **(Finite absorption)** There is a $N \in \mathbb{N}$ such that $\kappa_f^{\otimes N}(s, \cdot) = \delta_{s_f}$ for every $s \in \mathcal{S}$. We designate the corresponding DAG as *finitely absorbing*.

In this setting, the elements in the set $\mathcal{X} = \{s \in \mathcal{S} \setminus \{s_f\}\colon \kappa_f(s, \{s_f\}) > 0)\}$ are called *terminal states*. Illustratively, when $\mathcal{S}$ is finite and ν is the counting measure, the preceding definition

corresponds to a connected DAG with an edge from $s \in \mathcal{S}$ to $s' \in \mathcal{S}$ iff $\kappa_f(s, \{s'\}) > 0$, with condition 5) ensuring acyclicity and condition 2) implying connectivity. A GFlowNet, then, is characterized by a measurable pointed DAG, a potentially unnormalized distribution over terminal states $\mathcal{X}$ and learnable transition kernels on $\mathcal{S}$ (Definition 2). Realistically, its goal is to find an IGP over $\mathcal{S}$ which, starting at s_o, samples from $\mathcal{X}$ proportionally to a given positive function.

Definition 2 (GFlowNets [48])**.** A *GFlowNet* is a tuple $(\mathcal{G}, P_F, P_B, \mu)$ composed of a measurable pointed DAG $\mathcal{G}$, a σ-finite measure $\mu \ll \nu$, and σ-finite Markov kernels $P_F \ll \kappa_f$ and $P_B \ll \kappa_b$, respectively called *forward* and *backward* policies.

Training GFlowNets. In practice, we denote by $p_{F_\theta} \colon \mathcal{S} \times \mathcal{S} \to \mathbb{R}_+$ the density of P_F relative to κ_f, which we parameterize using a neural network with weigths θ. Similarly, we denote by p_B the density of P_B wrt k_b. Our objective is, for a given *target measure* $R \ll \mu$ on $\mathcal{X}$ with $r = {\mathrm{d}R}/{\mathrm{d}\mu}$, estimate the θ for which the distribution over $\mathcal{X}$ induced by $P_F(s_o, \cdot)$ matches R, i.e., for every $B \in \Sigma$,

$$\sum\nolimits_{n \geq 0} \int_{\mathcal{S}^n} p_{F_\theta}^{\otimes n}(s_o, s_{1:n}, s_f) \mathbb{1}_{s_n \in B} \kappa_f^{\otimes n}(s_o, \mathrm{d}s_{1:n}) = \frac{R(B)}{R(\mathcal{X})}.$$

Importantly, the above sum contains only finitely many non-zero terms due to the finite absorption property of κ_f. To ensure that p_{F_θ} abides by this equation, Lahlou et al. [48] showed it suffices that one of the next *balance conditions* are concomitantly satisfied by P_F and P_B.

Definition 3 (Trajectory balance condition)**.** For all $n \geq 0$ and $\mu^{\otimes n}$-almost surely $\forall s_{1:n} \in \mathcal{S}^n$, $p_{F_\theta}^{\otimes n}(s_o, s_{1:n}, s_f) = \frac{r(s_n)}{Z_\theta} p_B^{\otimes n}(s_n, s_{n:1}, s_o)$, w/ Z_θ denoting the target distribution's partition function.

Definition 4 (Detailed balance condition)**.** For an auxiliary parametric function $u \colon \mathcal{S} \to \mathbb{R}_+$ and $\mu^{\otimes 2}$-almost surely on $(s, s') \in \mathcal{S}^2$, $u(s) p_{F_\theta}(s, s') = u(s') p_B(s', s)$ and $u(x) = r(x)$ for $x \in \mathcal{X}$.

To enforce the trajectory balance (TB) or detailed balance (DB) conditions, we conventionally define a stochastic optimization problem to minimize the expected log-squared difference between the left- and right-hand sides of the corresponding condition under a probability measure ξ supported on an appropriate space [4, 15, 48, 49, 52, 55, 67]. For TB, e.g., we let ξ be defined on $\Sigma^{\otimes N}$ with support $\mathrm{supp}(\mu^{\otimes N})$. Then, we estimate the GFlowNet's parameters θ by minimizing

$$\mathop{\mathbb{E}}_{\tau \sim \xi} \left[\left(\sum_{0 \leq i \leq N-1} \left(\log \frac{p_{F_\theta}(s_i, s_{i+1})}{p_B(s_{i+1}, s_i)} \right) + \log \frac{Z_\theta}{r(s_h)} \right)^2 \right] \tag{1}$$

with $\tau = s_{o:N}$ and $h = \max\{i \colon s_i \neq s_f\}$ being the last non-final state's index in τ. As denoted by the subscript on Z_θ, using the TB loss incurs learning the target's normalizing constant. While tuning p_B during training is also possible, the common practice is to keep it fixed.

Henceforth, we will consider the measurable space of *trajectories* $(\mathcal{P}_\mathcal{S}, \Sigma_P)$, with $\mathcal{P}_\mathcal{S} = \{(s, s_1, \ldots, s_n, s_f) \in \mathcal{S}^{n+1} \times \{s_f\} \colon 0 \leq n \leq N-1\}$ and Σ_P as the σ-algebra generated by $\bigcup_{n=1}^{N+1} \Sigma^{\otimes n}$. For notational convenience, we use the same letters for representing the measures and kernels of $(\mathcal{S}, \Sigma)$ and their natural product counterparts in $(\mathcal{P}_\mathcal{S}, \Sigma_P)$, which exist by Carathéodory extension's theorem [96]; for example, $\nu(B) = \nu^{\otimes n}(B)$ for $B = (B_1, \ldots, B_n) \in \Sigma^{\otimes n}$ and $p_{F_\theta}(\tau | s_o; \theta)$ is the density of $P_F^{\otimes n+1}(s_o, \cdot)$ for $\tau = (s_o, s_1, \ldots, s_n, s_f)$ relatively to $\mu^{\otimes n}$. In this case, we will write τ for a generic element of $\mathcal{P}_\mathcal{S}$ and x for its terminal state (which is unique by Definition 1). For a comprehensive overview of GFlowNets, please refer to [48, 55].

GFlowNets and VI. GFlowNets can be interpreted as hierarchical variational models by framing the forward policy $p_{F_\theta}(\tau | s_o; \theta)$ in $(\mathcal{P}_\mathcal{S}, \Sigma_P)$ as a proposal to $\frac{r(x)}{Z} p_B(\tau | x)$. Malkin et al. [56] demonstrated that, for discrete target distributions, the TB loss in (1) aligns with the KL divergence in terms of expected gradients. Extending this, our Proposition 1 establishes that this relationship also holds for distributions over arbitrary topological spaces.

Proposition 1 (TB loss- and KL divergence gradients for topological spaces)**.** *Let* $\mathcal{L}_{TB}(\tau; \theta) = \left(\log {Z p_{F_\theta}(\tau | s_o; \theta)}/{r(x) p_B(\tau | x)}\right)^2$ *and* $p_B(\tau) = \frac{r(x)}{Z} p_B(s_{n-1:o} | x)$ *for* $\tau = (s_o, \ldots, s_{n-1}, x, s_f)$*. Then,*

$$\nabla_\theta \mathbb{E}_{\tau \sim P_F(s_o, \cdot)} \left[\mathcal{L}_{TB}(\tau; \theta) \right] = 2 \nabla_\theta \mathcal{D}_{KL}[P_F || P_B], \tag{2}$$

where $\mathcal{D}_{KL}[p_{F_\theta} || p_B] = \mathbb{E}_{\tau \sim P_F(s_o, \cdot)} \left[\log {p_{F_\theta}(\tau | s_o; \theta)}/{p_B(\tau)} \right]$ *is the KL divergence between* P_F *and* P_B*.*

https://doi.org/10.52202/079017-2416

This proposition shows that minimizing the on-policy TB loss is theoretically comparable to minimizing the KL divergence between P_F and P_B in terms of convergence speed. Since the TB loss requires estimating the intractable $R(\mathcal{X})$, the KL divergence, which avoids this estimation, can be a more suitable objective. Our experiments in Section 5 support this, with proofs provided in Appendix C. Extending this result to general topological spaces broadens the scope of divergence minimization strategies, extending guarantees for discrete spaces to continuous and mixed spaces. This generalization aligns with advances in generalized Bayesian inference [45] and generalized VI in function spaces [95], via optimization of generic divergences. We make the method theoretically firm and potentially widely applicable by proving the equivalence in these broader contexts.

Variance reduction. A naive Monte Carlo estimator for the gradient in Equation 2 has high variance [19], impacting the efficiency of stochastic gradient descent [97]. To mitigate this, we use *control variates*—random variables with zero expectation added to reduce the estimator's variance without bias [63, 71]. This method, detailed in Section 4, significantly reduces noise in gradient estimates and pragmatically improves training convergence, as shown in the experiments in Section 5.

3 Divergence measures for learning GFlowNets

This Section presents four different divergence measures for training GFlowNets and the accompanying gradient estimators for stochastic optimization. Regardless of the learning objective, recall that our goal is to estimate θ by minimizing a discrepancy measure D between P_F and P_B that is globally minimized if and only if $P_F = P_B$, i.e.,

$$\theta^* = \arg\min_{\theta} D(P_F, P_B), \tag{3}$$

in which P_B is typically fixed and P_F's density p_F^θ is parameterized by θ.

3.1 Renyi-α and Tsallis-α divergences

Renyi-α [74] and Tsallis-α [91] are families of statistical divergences including, as limiting cases, the widespread KL divergence (Section 3.2) [58]; see Definition 5. These divergences have been successfully applied to both variational inference [51] and policy search for model-based reinforcement learning [18]. Moreover, as we highlight in Section 5, their performance is competitive with, and sometimes better than, traditional learning objectives for GFlowNets based on minimizing log-squared differences between proposal and target distributions.

Definition 5 (Renyi-α and Tsallis-α divergences)**.** Let $\alpha \in \mathbb{R}$. Also, let p_{F_θ} and p_B be GFlowNet's forward and backward policies, respectively. Then, the *Renyi-α divergence* between P_F and P_B is

$$\mathcal{R}_\alpha(P_F||P_B) = \frac{1}{\alpha - 1} \log \int_{\mathcal{P}_\mathcal{S}} p_{F_\theta}(\tau|s_o)^\alpha p_B(\tau)^{1-\alpha} \kappa_f(s_o, \mathrm{d}\tau).$$

Similarly, the *Tsallis-α divergence* between P_F and P_B is

$$\mathcal{T}_\alpha(P_F||P_B) = \frac{1}{\alpha - 1} \left(\int_{\mathcal{P}_\mathcal{S}} p_{F_\theta}(\tau|s_o)^\alpha p_B(\tau)^{1-\alpha} \kappa_f(s_o, \mathrm{d}\tau) - 1 \right).$$

From Definition 5, we see that both Renyi-α and Tsallis-α divergences transition from a mass-covering to a mode-seeking behavior as α ranges from $-\infty$ to ∞. Regarding GFlowNet-training, this flexibility suggests that $\mathcal{R}_\alpha$ and $\mathcal{T}_\alpha$ are appropriate choices both, e.g., for carrying out Bayesian inference [16] — where interest lies in obtaining an accurate approximation to a posterior distribution —, and for combinatorial optimization [109] — where the goal is to find a few high-valued samples. Additionally, the choice of α provides a mechanism for controlling which trajectories are preferentially sampled during training, with larger values favoring the selection of trajectories leading to high-probability terminal states, resembling the effect of ε-greedy [55], thompson-sampling [73], local-search [40], and forward-looking [65, 66] techniques for carrying out off-policy training of GFlowNets [56].

To illustrate the effect of α on the learning dynamics of GFlowNets, we show in Figure 1 an *early stage* of training to sample from a homogeneous mixture of Gaussian distributions by minimizing Renyi-α divergence for different values of α; see Section 5.1 for details on this experiment. At this stage, we note that the GFlowNet covers the target distribution's modes but fails to separate them when α is large and negative. In contrast, a large positive α causes the model to focus on a single

high-probability region. Therefore, the use of an intermediate value for $\alpha = 0.5$ culminates in a model that accurately approximates the target distribution. Also, our early experiments suggested the persistence of such dependence on α for diverse learning tasks, with $\alpha = 0.5$ leading to the best results. Thus, we fix $\alpha = 0.5$ throughout our experimental campaign.

Importantly, we need only the gradients of $\mathcal{R}_\alpha$ and $\mathcal{T}_\alpha$ for solving the optimization problem in Equation 3 and, in particular, learning the target distribution's normalizing constant is unnecessary, as we underline in the lemma below. This property distinguishes such divergence measures from both TB and DB losses in Equation 1 and, in principle, simplifies the training of GFlowNets.

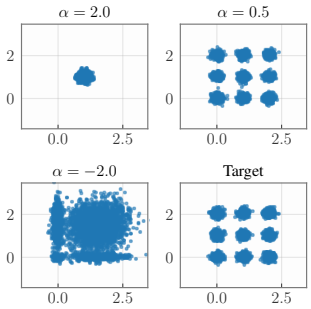

Figure 1: Mode-seeking ($\alpha = 2$) versus mass-covering ($\alpha = -2$) behaviour in α-divergences.

Lemma 1 (Gradients for $\mathcal{R}_\alpha$ and $\mathcal{T}_\alpha$)**.** *Let θ be the parameters of p_{F_θ} in Definition 5 and, for $\tau \in \mathcal{P}_\mathcal{S}$, $g(\tau, \theta) = \left({}^{p_B(\tau|x)r(x)}/_{p_{F_\theta}(\tau|s_o;\theta)}\right)^{1-\alpha}$. The gradient of $\mathcal{R}_\alpha$ wrt θ is*

$$\nabla_\theta \mathcal{R}_\alpha(p_{F_\theta}||p_B) = \frac{\mathbb{E}[\nabla_\theta g(\tau,\theta) + g(\tau,\theta)\nabla_\theta \log p_{F_\theta}(\tau|s_o;\theta)]}{(\alpha-1)\mathbb{E}[g(\tau,\theta)]};$$

the expectations are computed under P_F. Analogously, the gradient of $\mathcal{T}_\alpha$ wrt θ is

$$\nabla_\theta \mathcal{T}_\alpha(p_{F_\theta}||p_B) \overset{C}{=} \frac{\mathbb{E}[\nabla_\theta g(\tau,\theta) + g(\tau,\theta)\nabla_\theta \log p_{F_\theta}(\tau|s_o;\theta)]}{(\alpha-1)},$$

in which $\overset{C}{=}$ denotes equality up to a multiplicative constant.

Lemma 1 uses the REINFORCE method [97] to compute the gradients of both $\mathcal{R}_\alpha$ and $\mathcal{T}_\alpha$, and we implement Monte Carlo estimators to approximate the ensuing expectations based on a batch of trajectories $\{\tau_1, \ldots, \tau_N\}$ sampled during training [56]. Also, note that the function g is computed outside the log domain and, therefore, particular care is required to avoid issues such as numerical underflow of the unnormalized distribution [3, 87]. In our implementation, we sample an initial batch of trajectories $\{\tau_i\}_{i=1}^N$ and compute the maximum of r among the sampled terminal states in log space, i.e., $\log \tilde{r} = \max_i \log r(x_i)$. Then, we consider $\log \tilde{r}(x) = \log r(x) - \log \tilde{r}$ as the target's unnormalized log density. In Section 4, we will consider the design of variance reduction techniques to decrease the noise level of gradient estimates and possibly speed up the learning process.

3.2 Kullback-Leibler divergence

The KL divergence [47] is a limiting member of the Renyi-α and Tsallis-α families of divergences, derived when $\alpha \to 1$ [70], and is the most widely deployed divergence measure in statistics and machine learning. To conduct variational inference, one regularly considers both the *forward* and *reverse* KL divergences, which we review in the definition below.

Definition 6 (Forward and reverse KL)**.** The *forward* KL divergence between a target P_B and a proposal P_F is $\mathcal{D}_{KL}[P_B||P_F] = \mathbb{E}_{\tau\sim P_B(s_f,\cdot)}\left[\log {}^{p_B(\tau)}/_{p_{F_\theta}(\tau|s_o)}\right]$. Also, the *reverse* KL divergence is defined by $\mathcal{D}_{KL}[P_F||P_B] = \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[\log {}^{p_{F_\theta}(\tau|s_o)}/_{p_B(\tau)}\right]$.

Remarkably, we cannot use a simple Monte Carlo estimator to approximate the forward KL due to the presumed intractability of P_B (which depends directly on R). As a first approximation, we could estimate $\mathcal{D}_{KL}[P_B||P_F]$ via importance sampling w/ P_F as a proposal distribution as in [56]:

$$\mathcal{D}_{KL}[P_B||P_F] = \mathbb{E}_{\tau\sim P_F}\left[\frac{p_B(\tau)}{p_{F_\theta}(\tau|s_o)}\log\frac{p_B(\tau)}{p_{F_\theta}(\tau|s_o)}\right], \tag{4}$$

and subsequently implement a REINFORCE estimator to compute $\nabla_\theta \mathcal{D}_{KL}[P_B||P_F]$. Nevertheless, as we only need the divergence's derivatives to perform SGD, we apply the importance weights directly to the gradient estimator. We summarize this approach in the lemma below.

Lemma 2 (Gradients for the KL divergence)**.** *Let θ be the parameters of P_F and $s(\tau;\theta) = \log p_{F_\theta}(\tau|s_o;\theta)$. Then, the gradient of $\mathcal{D}_{KL}[P_F||P_B]$ relatively to θ satisfies*

$$\nabla_\theta \mathcal{D}_{KL}\left[P_F||P_B\right] = \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[\nabla_\theta s(\tau;\theta) + \log\frac{p_{F_\theta}(\tau|s_o)}{p_B(\tau|x)r(x)}\nabla_\theta s(\tau;\theta)\right]$$

https://doi.org/10.52202/079017-2416

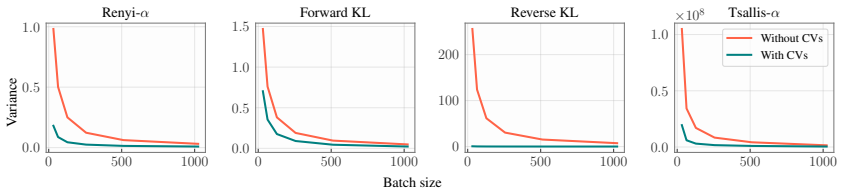

Figure 2: **Variance of the estimated gradients as a function of the trajectories' batch size.** Our control variates greatly reduce the estimator's variance, even for relatively small batch sizes.

Correspondingly, the gradient of $\mathcal{D}_{KL}[P_B||P_F]$ *wrt* θ *is*

$$\nabla_\theta \mathcal{D}_{KL}[P_B||P_F] \overset{C}{=} -\mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[\frac{p_{F_\theta}(\tau|s_o)}{p_B(\tau|x)r(x)}\nabla_\theta s(\tau;\theta)\right].$$

Crucially, choosing an appropriate learning objective is an empirical question that one should consider on a problem-by-problem basis — similar to the problem of selecting among Markov chain simulation techniques [27]. In particular, a one-size-fits-all solution does not exist; see Section 5 for a thorough experimental investigation. Independently of the chosen method, however, the Monte Carlo estimators for the quantities outlined in Lemma 2 are of potentially high variance and may require a relatively large number of trajectories to yield a reliable estimate of the gradients [97]. The following sections demonstrate that variance reduction techniques alleviate this issue.

4 Control variates for low-variance gradient estimation

Control variates. We first review the concept of a control variate. Let $f\colon \mathcal{P}_\mathcal{S} \to \mathbb{R}$ be a real-valued measurable function and assume that our goal is to estimate $\mathbb{E}_{\tau\sim\pi}[f(\tau)]$ according to a probability measure π on Σ_P (see Section 2 to recall the definitions). The variance of a naive Monte Carlo estimator for this quantity is $\mathrm{Var}_\pi(f(\tau))/n$. On the other hand, consider a random variable (RV) $g\colon \mathcal{P}_\mathcal{S} \to \mathbb{R}$ positively correlated with f and with known expectation $\mathbb{E}_\pi[g(\tau)]$. Then, the variance of a naive Monte Carlo for $\mathbb{E}_\pi[f(\tau) - a(g(\tau) - \mathbb{E}_\pi[g(\tau)])]$ for a *baseline* $a \in \mathbb{R}$ is

$$\frac{1}{n}\left[\mathrm{Var}_\pi(f(\tau)) - 2a\mathrm{Cov}_\pi(f(\tau), g(\tau)) + a^2\mathrm{Var}_\pi(g(\tau))\right], \tag{5}$$

which is potentially smaller than $\frac{1}{n}\mathrm{Var}_\pi(f(\tau))$ if the covariance between f and g is sufficiently large. Under these conditions, we choose the value of a that minimizes Equation 5 [94], namely, $a = \mathrm{Cov}_\pi(f(\tau),g(\tau))/\mathrm{Var}_\pi(g(\tau))$. We then call the function g a *control variate* [63]. Also, although the quantities defining the best baseline a are generally unavailable in closed form, one commonly uses a batch-based estimate of $\mathrm{Cov}_\pi(f(\tau), g(\tau))$ and $\mathrm{Var}_\pi(g(\tau))$; the incurred bias is generally negligible relatively to the reduced variance [71, 79, 85]. For vector-valued RVs, we let a be a diagonal matrix and exhibit, in the next proposition, the optimal baseline minimizing the covariance matrix's trace.

Proposition 2 (Control variate for gradients)**.** *Let* $f, g\colon \mathcal{P}_\mathcal{S} \to \mathbb{R}^d$ *be vector-valued functions and* π *be a probability measure on* $\mathcal{P}_\mathcal{S}$*. Consider a* baseline $a \in \mathbb{R}$ *and assume* $\mathbb{E}_\pi[g(\tau)] = 0$*. Then,*

$$\arg\min_{a\in\mathbb{R}} \mathrm{Tr}\,\mathrm{Cov}_\pi[f(\tau) - a\cdot g(\tau)] = \frac{\mathbb{E}_\pi[g(\tau)^T(f(\tau) - \mathbb{E}_\pi[f(\tau')])]}{\mathbb{E}_\pi[g(\tau)^T g(\tau)]}.$$

Note that, when implementing the REINFORCE gradient estimator, the expectation we wish to estimate may be generally written as $\mathbb{E}_{P_F(s_o,\cdot)}[\nabla_\theta f(\tau) + f(\tau)\nabla_\theta \log p_{F_\theta}(\tau)]$. For the second term, we use a leave-one-out estimator [85]; see below. For the first term, we use $\nabla_\theta \log p_{F_\theta}$ as a control variate, which satisfies $\mathbb{E}_{P_F(s_o,\cdot)}[\nabla_\theta \log p_{F_\theta}(\tau|s_o;\theta)] = 0$. Importantly, estimating the optimal baseline $a^\star$ in Proposition 2 cannot be done efficiently due to the non-linear dependence of the corresponding Monte Carlo estimator on the sample-level gradients [2]; i.e., it cannot be represented as a vector-Jacobian product, which is efficient to compute in reverse-mode automatic differentiation (*autodiff*) frameworks [8, 69]. Consequently, we consider a linear approximation of both numerator and denominator defining $a^\star$ in Proposition 2, which may be interpreted as an instantiation of the

delta method [83, Sec. 7.1.3]. Then, given a batch $\{\tau_1, \dots, \tau_N\}$ of trajectories, we instead use

$$\hat{a} = \frac{\left\langle \sum_{n=1}^{N} \nabla_\theta \log p_{F_\theta}(\tau_n), \sum_{n=1}^{N} \nabla_\theta f(\tau_n) \right\rangle}{\epsilon + \left\| \sum_{n=1}^{N} \nabla_\theta \log p_{F_\theta}(\tau_n) \right\|^2} \tag{6}$$

as the REINFORCE batch-based estimated baseline; $\langle \cdot, \cdot \rangle$ represents the inner product between vectors. Intuitively, the numerator is roughly a linear approximation to the covariance between $\nabla_\theta \log p_{F_\theta}$ and $\nabla_\theta f$ under P_F. In contrast, the denominator approximately measures the variance of $\nabla_\theta \log p_{F_\theta}$, and $\epsilon > 0$ is included to enhance numerical stability. As a consequence, for the reverse KL divergence, $\nabla_\theta f(\tau) = \nabla_\theta \log p_{F_\theta}(\tau)$, $\hat{a} \approx 1$ and the term corresponding to the expectation of $\nabla_\theta f(\tau)$ vanishes. We empirically find that this approach frequently reduces the variance of the estimated gradients by a large margin (see Figure 2 above and Section 5 below).

Leave-one-out estimator. We now focus on obtaining a low-variance estimate of $\mathbb{E}_{\tau \sim P_F(s_o, \cdot)}[f(\tau) \nabla_\theta \log p_{F_\theta}(\tau)]$. As an alternative to the estimator of Proposition 2, Shi et al. [85] and Salimans and Knowles [82] proposed a sample-dependent baseline of the form $a(\tau_i) = \frac{1}{N-1} \sum_{1 \le n \le N, n \neq i} f(\tau_n)$ for $i \in \{1, \dots, N\}$. The resulting estimator,

$$\delta = \frac{1}{N} \sum_{n=1}^{N} \left(f(\tau_n) - \frac{1}{N-1} \sum_{j=1, j \neq i}^{N} f(\tau_j) \right) \nabla_\theta \log p_{F_\theta}(\tau_n),$$

is unbiased for $\mathbb{E}\left[f(\tau) \nabla_\theta \log p_{F_\theta}(\tau)\right]$ due to the independence between τ_i and τ_j for $i \neq j$. Strikingly, δ can be swiftly computed with *autodiff*: if $\mathbf{f} = (f(\tau_n))_{n=1}^N$ and $\mathbf{p} = (\log p_{F_\theta}(\tau_n))_{n=1}^N$, then

$$\delta = \nabla_\theta \frac{1}{N} \left\langle \mathrm{sg}\left(\mathbf{f} - \frac{1}{N-1} (\mathbf{1} - \mathbf{I}) \mathbf{f} \right), \mathbf{p} \right\rangle, \tag{7}$$

with sg as the stop-gradient operation (e.g., `lax.stop_gradient` in JAX [8] and `torch.detach` in PyTorch [69]). Importantly, these techniques incur a minimal computational overhead to the stochastic optimization algorithms relative to the considerable reduction in variance they enact.

Relationship with previous works. Importantly, Malkin et al. [56] used $\hat{a} = \frac{1}{N} \sum_n f(\tau_n)$ as baseline and an importance-weighted aggregation to adjust for the off-policy sampling of trajectories, introducing bias in the gradient estimates and relinquishing guarantees of the optimization procedure. A learnable baseline independently trained to match $\hat{b}$ was also considered. This potentially entailed the inaccurate conclusion that the TB and DB are superior to standard divergence-based objectives. Indeed, the following section underlines that such divergence measures are sound and practical learning objectives for GFlowNets for a range of tasks.

Illustration of the control variates' effectiveness. We train the GFlowNets using increasingly larger batches of $\{2^i : i \in [[5, 10]]\}$ trajectories with and without CVs. In this setting, Figure 2 showcases the drastic reduction in the variance, represented by the covariance matrix's trace, of the estimated learning objectives' gradients w.r.t. the model's parameters promoted by the CVs. Impressively, as we show in Figure 6, this approach significantly increases the efficiency of the underlying stochastic optimization algorithm. See Section 5 and Appendix D for further details.

5 Training GFlowNets with divergence measures

Our experiments serve two purposes. Firstly, we show in Section 5.2 that minimizing divergence-based learning objectives leads to competitive and often better approximations than the alternatives based on log-squared violations of the flow network's balance. This underlines the effectiveness of well-established divergence measures for training GFlowNets [56, 112]. Secondly, we highlight in Section 5.3 that the reduction of variance enacted by our control variates critically accelerates the convergence of GFlowNets. We consider widely adopted benchmark tasks from GFlowNet literature, described in Section 5.1, contemplating both discrete and continuous target distributions. Please refer to Appendix B and Appendix E for additional information on the experimental setup.

5.1 Generative tasks

Below, we provide a high-level characterization of the generative tasks used for synthetic data generation and training. For a more rigorous description in the light of Section 2, see Appendix B.

https://doi.org/10.52202/079017-2416

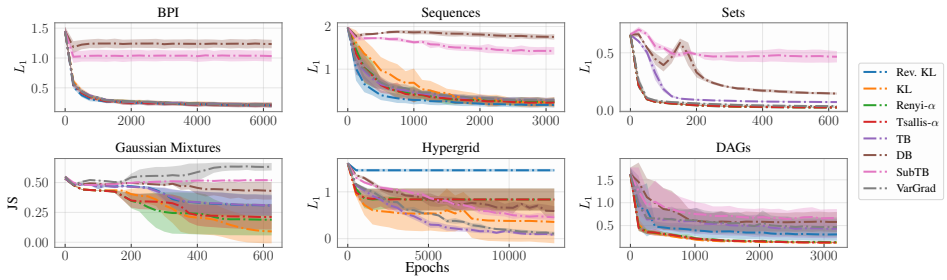

Figure 3: **Divergence-based learning objectives often lead to faster training than TB loss.** Notably, contrasting with the experiments of [56], there is no single best loss function always conducting to the fastest convergence rate, and minimizing well-known divergence measures is often on par with or better than minimizing the TB loss in terms of convergence speed. Results were averaged across three different seeds. Also, we fix $\alpha = 0.5$ for both Tsallis-α and Renyi-α divergences.

Set generation [3, 34, 65, 66]**.** A state s corresponds to a set of size up to a given S and the terminal states $\mathcal{X}$ are sets of size S; a transition corresponds to adding an element from a deposit $\mathcal{D}$ to s. The IGP starts at an empty set, and the log-reward of a $x \in \mathcal{X}$ is $\sum_{d \in x} f(d)$ for a fixed $f : \mathcal{D} \to \mathbb{R}$.

Autoregressive sequence generation [32, 55]**.** Similarly, a state is a seq. s of max size S and a terminal state is a seq. ended by an end-of-sequence token; a transition appends $d \in \mathcal{D}$ to s. The IGP starts with an empty sequence and, for $x \in \mathcal{X}$, $\log r(x) = \sum_{i=1\ldots|x|} g(i) f(x_i)$ for functions f, g.

Bayesian phylogenetic inference (BPI) [111]**.** A state s is a forest composed of binary trees with labeled leaves and unlabelled internal nodes, and a transition amounts to joining the roots of two trees to a newly added node. Then, s is terminal when it is a single connected tree — called a *phylogenetic tree*. Finally, given a dataset of nucleotide sequences, the reward function is the unnormalized posterior over trees induced by J&C69's mutation model [37] and a uniform prior.

Hypergrid navigation [3, 55, 56, 66]**.** A state $s \in \{0, \ldots, H-1\}^d$ is a component of a H-sized and d-dimensional Euclidean grid. The IGP starts at $\mathbf{0}$ and, if we let δ_i be the i-th line of the identity matrix and $\Delta(s) = \{\delta_i : i \in \{1, \ldots, d\} \wedge \max_j (s + \delta_i)_j < H\}$, a transition either adds a $\delta \in \Delta(s)$ to s or stops at s. We use Malkin et al. [55, Section 5.1]'s reward function with $R_o = 10^{-3}$.

Bayesian structure learning [15, 16]**.** A state s is a DAG representing a Bayesian network; a transition either adds an edge to s or stops the IGP. Similarly to Deleu et al. [15], we ensure the added edges preserve the state's acyclicity. The reward function is defined as the maximum likelihood of the linear Gaussian structural model induced by the current state based on a fixed i.i.d. data set.

Mixture of Gaussians (GMs) [48, 110]**.** The IGP starts at $\mathbf{0} \in \mathbb{R}^d$ and proceeds by sequentially substituting each coordinate with a sample from a real-valued distribution. For a K-component GM, the reward of $\mathbf{x} \in \mathbb{R}^d$ is defined as $\sum_{k=} \alpha_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$ with $\alpha_k \geq 0$ and $\sum_k \alpha_k = 1$.

Banana-shaped distribution [57, 76]**.** We use the same IGP implemented for a bi-dimensional GM. For $\mathbf{z} \in \mathbb{R}^2$, we set $r(\mathbf{x})$ to a normal likelihood defined on a quadratic function of $\mathbf{x}$, see Equation 8 in the supplement. We use HMC samples as ground truth to gauge performance on this task.

5.2 Assessing convergence speed

Next, we provide evidence that minimizing divergence-based objectives frequently leads to faster convergence than minimizing the standard TB loss [55].

Experimental setup. We compare the convergence speed in terms of the rate of decrease of a measure of distributional error when using different learning objectives for a GFlowNet trained to sample from each of the distributions described in Section 5.1. For discrete distributions, we adopt the evaluation protocols of previous works [3, 54, 55, 66] and compute the L_1 distance between the learned $p_T(x; \theta)$ and target $r(x)$, namely, $\sum_{x \in \mathcal{X}} |p_T(x; \theta) - {}^{r(x)}/_{Z}|$. To approximate p_T, we use a Monte Carlo estimate of $p_T(x; \theta) = \mathbb{E}_{\tau \sim P_B(x, \cdot)} \left[{}^{p_{F_\theta}(\tau | s_o; \theta)}/_{p_B(\tau | x)} \right]$. For continuous distributions, we echo [48, 110] and compute Jensen-Shannon's divergence between $P_T(x; \theta)$ and $R(x)$:

$$\mathcal{D}_{JS}[P_T || R] = {}^1/_2 \left(\mathcal{D}_{KL}[P_T || M] + \mathcal{D}_{KL}[R || M] \right) = \mathop{\mathbb{E}}_{x \sim P_T} \left[\log {}^{p_T(x)}/_{m(x)} \right] + \mathop{\mathbb{E}}_{x \sim R} \left[\log {}^{r(x)}/_{Z m(x)} \right],$$

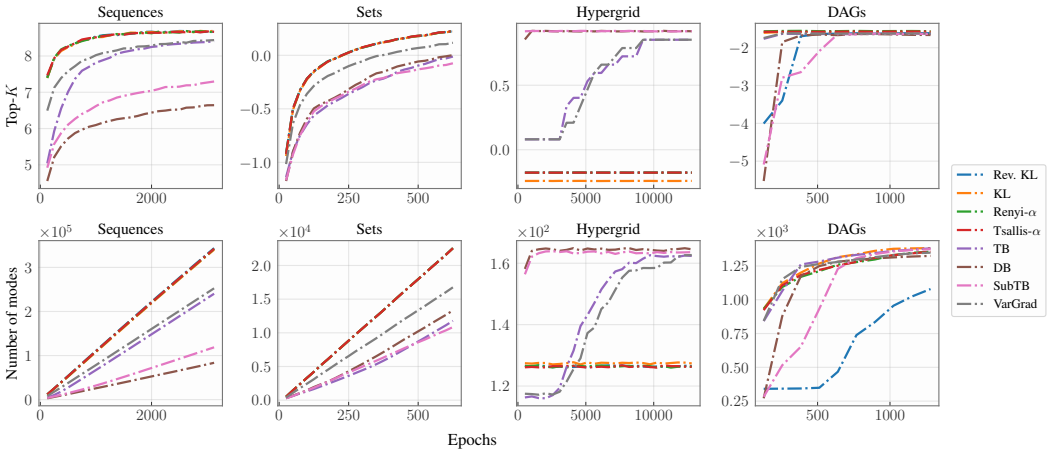

Figure 4: **Average reward for the K highest scoring samples (top-K) and Number of Modes** found during training for the tasks of sequence design, set generation, hypergrid and DAG environments. With the only exception of the hypergrid task, the minimization of divergence-based measures leads to similar and often faster discovery of high-valued states relatively to their balance-based counterparts.

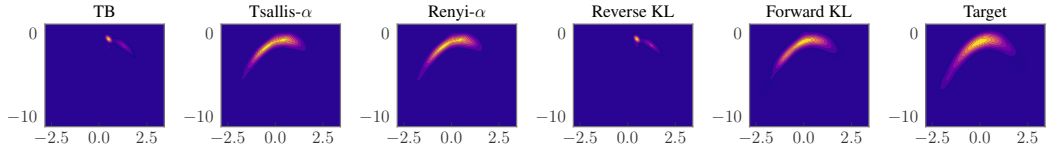

Figure 5: **Learned distributions for the banana-shaped target.** Tsallis-α, Renyi-α and for. KL leads to a better model than TB and Rev. KL, which behave similarly — as predicted by Proposition 1.

with $M(B) = 1/2\,(P_T(B) + {}^{R(B)}/_{R(\mathcal{X})})$ being the averaged measure of P_T and R and m its corresponding density relatively to the reference measure μ. Remarkably, for the GMs distribution, we can directly sample from the target to estimate $\mathcal{D}_{KL}[R||M]$, and the autoregressive nature of the generative process ensures that $p_T(x) = p_{F_\theta}(\tau|s_o)$ for the unique trajectory τ starting at s_o and finishing at x. Hence, we get an unbiased estimate of $\mathcal{D}_{KL}[P_T||M]$. Finally, let X_t be the first t terminal states encountered during training and $\{x_{(1)}, \ldots, x_{(k)}\}$ be an descending ordering of X_t according to r. Then, we select a threshold $\tau \in \mathbb{R}$ and an integer K to report $\mathrm{NoM}(\mathrm{X}_t) = |\{r(x) \colon r(x) \geq \tau \wedge x \in \mathrm{X}_t\}|$, called *number of modes*, and $\mathrm{TopK}(\mathrm{X}_t) = \mathrm{AVG}\left(\{r(x_{(i)}) \colon 1 \leq i \leq K\}\right)$, referred to as *top-K average reward*. Both NoM and TopK are metrics of substantial interest in the GFlowNet literature [3, 55, 56, 64, 65].

Results. Figure 3 shows that the procedure minimizing divergence-based measures accelerates the training convergence of GFlowNets, whereas Figure 5 (for the banana-shaped distribution) and Table 1 highlight that we obtain a more accurate model with a fix compute budget. The difference between learning objectives is not statistically significant for the BPI task. Also, we may attribute the superior performance of reverse KL compared to the forward in the sequence generation task to the high variance of the importance-sampling-based gradient estimates. Indeed, the observed differences disappear when we increase the batch of trajectories to reduce the estimator's variance (see Figure 8 in Appendix D). In conclusion, our empirical results based on experiments testing diverse generative settings and expanding prior art [48, 56, 112], shows that training methods based on minimizing f-divergence VI objectives with adequate CVs implemented are practical and effective in many tasks. Correlatively, Figure 4 supports

Table 1: Divergence minimization achieves better than or similar accuracy compared to enforcing TB.

	BPI	Sequences	Sets	GMs
TB	0.22±0.04	0.28±0.06	0.07±0.00	0.31±0.08
Rev. KL	0.21±0.04	**0.16**±0.06	**0.03**±0.00	0.31±0.09
For. KL	0.22±0.04	0.23±0.12	**0.03**±0.00	**0.09**±0.10
Renyi-α	0.22±0.03	0.23±0.10	**0.03**±0.00	0.19±0.13
Tsallis-α	0.21±0.04	0.22±0.09	**0.03**±0.00	0.21±0.11

https://doi.org/10.52202/079017-2416

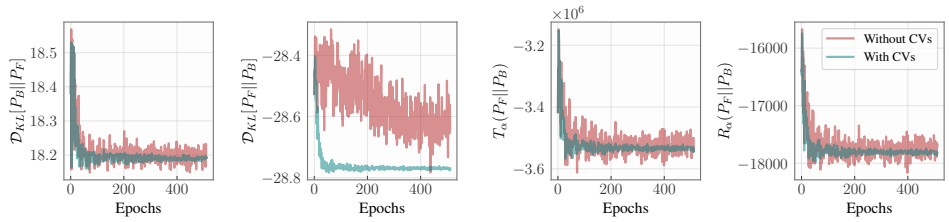

Figure 6: **Learning curves for different objective functions** in the task of set generation. The reduced variance of the gradient estimates notably increases training stability and speed.

the fact that minimizing divergence-based objectives frequently implies better coverage of the target's high-probability regions; the only exception is the (extremely sparse) hypergrid task [55].

5.3 Reducing the variance of the estimated gradients

Figure 2 demonstrates that implementing CVs for the REINFORCE estimator reduces the noise level of gradient estimates significantly. This reduction in variance also accelerates training convergence. To illustrate this, we use the same experimental setup from Section 5.2 and analyze the learning curves for each divergence measure with and without control variates.

Results. Figure 6 shows that the implemented gradient reduction techniques significantly enhance the learning stability of GFlowNets and drastically accelerate training convergence when minimizing the reverse KL divergence. Our results indicate that well-designed CVs for gradient estimation can greatly benefit GFlowNets training. Notably, similar improvements have been observed in the context of Langevin dynamics simulations [22, 31, 44] and policy gradient methods for RL [68, 100].

6 Conclusions, limitations and broader impact

Discussion. We showed in a comprehensive range of experiments that divergence measures common in VI — forward KL, reverse KL, Renyi-α, and Tsallis-α — are effective learning objectives for training GFlowNets, performing competitively with or better than their balance-based counterparts. To achieve this, the introduction of efficacious control variates for low-variance gradient estimation of the divergence-based objectives was crucial, which is a key distinction between our work an prior art [55, 112]. Additionally, we developed the theoretical connection between GFlowNets and VI beyond the setting of finitely supported measures, establishing results for arbitrary topological spaces.

Limitations. Albeit comprehensive and on par with the wider literature, our empirical evaluation was performed on problems of relatively small size due to the intractability of probing a GFlowNet's distributional accuracy on very large state spaces. That said, we acknowledge that an assessment on the domains of natural language processing [30] and drug discovery [3] based on context-specific metrics would strengthen our conclusions; we leave these tasks to future endeavors. Similarly, while we observed promising results for $\alpha = 0.5$, there might be different choices of α that, depending on the application, might strike a better explore-exploit tradeoff and incur faster convergence. Thus, thoroughly exploring different α might be especially useful to practitioners.

Broader impact. Overall, our work highlights the potential of the once-dismissed VI-inspired schemes for training GFNs, paving the way for further research towards improving the GFlowNets by drawing inspiration from the VI literature. For instance, one could develop χ-divergence-based losses for GFNs [19], combine GFNs with MCMC using Ruiz and Titsias [81]'s divergence, or employ an objective similar to that of importance-weighted autoencoders [10]. Finally, although an ϵ-greedy off-policy sampling scheme can be easily incorporated into a divergence-minimizing algorithm through an importance-sampling correction, it remains elusive whether this would be possible for more sophisticated sampling techniques such as replay buffer [15] and local search [40].

Acknowledgements

This work was supported by the Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro FAPERJ (SEI-260003/000709/2023), the São Paulo Research Foundation FAPESP (2023/00815-6), the Conselho Nacional de Desenvolvimento Científico e Tecnológico CNPq (404336/2023-0), and the Silicon Valley Community Foundation through the University Blockchain Research Initiative (Grant #2022-199610).

References

[1] S. Axler. *Measure, Integration &; Real Analysis*. Springer International Publishing, 2020. ISBN 9783030331436. doi: 10.1007/978-3-030-33143-6.

[2] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 2018.

[3] E. Bengio, M. Jain, M. Korablyov, D. Precup, and Y. Bengio. Flow network based generative models for non-iterative diverse candidate generation. In *NeurIPS (NeurIPS)*, 2021.

[4] Y. Bengio, S. Lahlou, T. Deleu, E. J. Hu, M. Tiwari, and E. Bengio. Gflownet foundations. *Journal of Machine Learning Research (JMLR)*, 2023.

[5] M. Betancourt. A conceptual introduction to hamiltonian monte carlo. *arXiv preprint arXiv:1701.02434*, 2017.

[6] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2007.

[7] D. M. Blei and et al. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 2017.

[8] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[9] L. Buesing, N. Heess, and T. Weber. Approximate inference in discrete distributions with monte carlo tree search and value functions. In *AISTATS*, pages 624–634. PMLR, 2020.

[10] Y. Burda, R. B. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. In *ICLR (Poster)*, 2016.

[11] P. Carbonetto, M. King, and F. Hamze. A stochastic approximation method for inference in probabilistic graphical models. *NeurIPS*, 2009.

[12] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 2017.

[13] T. da Silva, E. Silva, A. Ribeiro, A. Góis, D. Heider, S. Kaski, and D. Mesquita. Human-in-the-loop causal discovery under latent confounding using ancestral gflownets. *arXiv preprint:2309.12032*, 2023.

[14] M. P. Deisenroth and C. E. Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *ICML*, Proceedings of Machine Learning Research, pages 465–472. PMLR, 2011.

[15] T. Deleu, A. Góis, C. C. Emezue, M. Rankawat, S. Lacoste-Julien, S. Bauer, and Y. Bengio. Bayesian structure learning with generative flow networks. In *UAI*, 2022.

[16] T. Deleu, M. Nishikawa-Toomey, J. Subramanian, N. Malkin, L. Charlin, and Y. Bengio. Joint Bayesian inference of graphical structure and parameters with a single generative flow network. In *Advances in Neural Processing Systems (NeurIPS)*, 2023.

[17] T. Deleu, P. Nouri, N. Malkin, D. Precup, and Y. Bengio. Discrete probabilistic inference as control in multi-path environments. *CoRR*, abs/2402.10309, 2024.

[18] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft. Learning and policy search in stochastic dynamical systems with bayesian neural networks. *arXiv preprint arXiv:1605.07127*, 2016.

[19] A. B. Dieng, D. Tran, R. Ranganath, J. Paisley, and D. Blei. Variational inference via χ upper bound minimization. *NeurIPS*, 2017.

[20] J. Domke. Provable gradient variance guarantees for black-box variational inference. In *NeurIPS*, pages 328–337, 2019.

 https://doi.org/10.52202/079017-2416

[21] J. Domke. Provable smoothness guarantees for black-box variational inference. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 2587–2596. PMLR, 2020.

[22] K. A. Dubey, S. J. Reddi, S. A. Williamson, B. Poczos, A. J. Smola, and E. P. Xing. Variance reduction in stochastic gradient langevin dynamics. In *NeurIPS*, 2016.

[23] S. Dutta, M. Das, and U. Maulik. Towards causality-based explanation of aerial scene classifiers. *IEEE Geoscience and Remote Sensing Letters*, 2023.

[24] J.-P. Falet, H. B. Lee, N. Malkin, C. Sun, D. Secrieru, D. Zhang, G. Lajoie, and Y. Bengio. Delta-ai: Local objectives for amortized inference in sparse graphical models, 2023.

[25] J. Felsenstein. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 1981.

[26] T. Garipov, S. D. Peuter, G. Yang, V. Garg, S. Kaski, and T. S. Jaakkola. Compositional sculpting of iterative generative processes. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[27] C. J. Geyer. Markov chain monte carlo maximum likelihood. 1991.

[28] G. Hinton, N. Srivastava, and K. Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. 2012.

[29] E. J. Hu, N. Malkin, M. Jain, K. E. Everett, A. Graikos, and Y. Bengio. Gflownet-em for learning compositional latent variable models. In *International Conference on Machine Learning (ICLR)*, 2023.

[30] E. J. Hu, M. Jain, E. Elmoznino, Y. Kaddar, G. Lajoie, Y. Bengio, and N. Malkin. Amortizing intractable inference in large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[31] Z. Huang and S. Becker. Stochastic gradient langevin dynamics with variance reduction. *CoRR*, 2021.

[32] M. Jain, E. Bengio, A. Hernandez-Garcia, J. Rector-Brooks, B. F. P. Dossou, C. A. Ekbote, J. Fu, T. Zhang, M. Kilgour, D. Zhang, L. Simine, P. Das, and Y. Bengio. Biological sequence design with GFlowNets. In *International Conference on Machine Learning (ICML)*, 2022.

[33] M. Jain, T. Deleu, J. Hartford, C.-H. Liu, A. Hernandez-Garcia, and Y. Bengio. Gflownets for ai-driven scientific discovery. *Digital Discovery*, 2023.

[34] H. Jang, M. Kim, and S. Ahn. Learning energy decompositions for partial inference in GFlownets. In *The Twelfth International Conference on Learning Representations*, 2024.

[35] M. Järvenpää and J. Corander. On predictive inference for intractable models via approximate bayesian computation. *Statistics and Computing*, 33(2), Feb. 2023. ISSN 1573-1375.

[36] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233, 1999.

[37] T. H. Jukes and C. R. Cantor. Evolution of protein molecules. In *Mammalian Protein Metabolism*. Elsevier, 1969.

[38] H. J. Kappen, V. Gómez, and M. Opper. Optimal control as a graphical model inference problem. *Mach. Learn.*, 87(2):159–182, 2012.

[39] K. Kim, Y. Ma, and J. Gardner. Linear convergence of black-box variational inference: Should we stick the landing? In *AISTATS*, volume 238 of *Proceedings of Machine Learning Research*, pages 235–243. PMLR, 2024.

[40] M. Kim, T. Yun, E. Bengio, D. Zhang, Y. Bengio, S. Ahn, and J. Park. Local search gflownets. *arXiv preprint arXiv:2310.02710*, 2023.

[41] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[42] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling. Semi-supervised learning with deep generative models. *NeurIPS*, 2014.

[43] D. P. Kingma, T. Salimans, B. Poole, and J. Ho. Variational diffusion models, 2023.

[44] Y. Kinoshita and T. Suzuki. Improved convergence rate of stochastic gradient langevin dynamics with variance reduction and its application to optimization. In *NeurIPS*, 2022.

[45] J. Knoblauch, J. Jewson, and T. Damoulas. An optimization-centric view on bayes' rule: Reviewing and generalizing variational inference. *J. Mach. Learn. Res.*, 23:132:1–132:109, 2022.

[46] A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, and D. M. Blei. Automatic differentiation variational inference. *J. Mach. Learn. Res.*, 18:14:1–14:45, 2017.

[47] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 1951.

[48] S. Lahlou, T. Deleu, P. Lemos, D. Zhang, A. Volokhova, A. Hernández-García, L. N. Ezzine, Y. Bengio, and N. Malkin. A theory of continuous generative flow networks. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 18269–18300. PMLR, 2023.

[49] E. Lau, N. M. Vemgal, D. Precup, and E. Bengio. DGFN: Double generative flow networks. In *NeurIPS 2023 Generative AI and Biology (GenBio) Workshop*, 2023.

[50] S. Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *CoRR*, abs/1805.00909, 2018.

[51] Y. Li and R. E. Turner. Rényi divergence variational inference. *NeurIPS*, 29, 2016.

[52] D. Liu and et al. Gflowout: Dropout with generative flow networks. In *International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[53] R. Liu, J. Regier, N. Tripuraneni, M. I. Jordan, and J. D. McAuliffe. Rao-blackwellized stochastic gradients for discrete distributions. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 4023–4031. PMLR, 2019.

[54] K. Madan, J. Rector-Brooks, M. Korablyov, E. Bengio, M. Jain, A. C. Nica, T. Bosc, Y. Bengio, and N. Malkin. Learning gflownets from partial episodes for improved convergence and stability. In *International Conference on Machine Learning*, 2022.

[55] N. Malkin, M. Jain, E. Bengio, C. Sun, and Y. Bengio. Trajectory balance: Improved credit assignment in GFlownets. In *NeurIPS (NeurIPS)*, 2022.

[56] N. Malkin, S. Lahlou, T. Deleu, X. Ji, E. Hu, K. Everett, D. Zhang, and Y. Bengio. GFlowNets and variational inference. *International Conference on Learning Representations (ICLR)*, 2023.

[57] D. Mesquita, P. Blomstedt, and S. Kaski. Embarrassingly parallel MCMC using deep invertible transformations. In *UAI*, 2019.

[58] T. Minka. Divergence measures and message passing. Technical report, Research, Microsoft, 2005. URL https://www.seas.harvard.edu/courses/cs281/papers/minka-divergence.pdf.

[59] S. Mohamed, M. Rosca, M. Figurnov, and A. Mnih. Monte carlo gradient estimation in machine learning. *J. Mach. Learn. Res.*, 21:132:1–132:62, 2020.

[60] S. Mohammadpour, E. Bengio, E. Frejinger, and P.-L. Bacon. Maximum entropy gflownets with soft q-learning, 2023.

 https://doi.org/10.52202/079017-2416

[61] R. M. Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2011.

[62] A. C. Nica, M. Jain, E. Bengio, C.-H. Liu, M. Korablyov, M. M. Bronstein, and Y. Bengio. Evaluating generalization in gflownets for molecule design. In *ICLR2022 Machine Learning for Drug Discovery*, 2022.

[63] A. B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[64] L. Pan, N. Malkin, D. Zhang, and Y. Bengio. Better training of GFlowNets with local credit and incomplete trajectories. In *International Conference on Machine Learning (ICML)*, 2023.

[65] L. Pan, N. Malkin, D. Zhang, and Y. Bengio. Better training of gflownets with local credit and incomplete trajectories. *arXiv preprint arXiv:2302.01687*, 2023.

[66] L. Pan, D. Zhang, A. Courville, L. Huang, and Y. Bengio. Generative augmented flow networks. In *International Conference on Learning Representations (ICLR)*, 2023.

[67] L. Pan, D. Zhang, M. Jain, L. Huang, and Y. Bengio. Stochastic generative flow networks. In *UAI*, volume 216 of *Proceedings of Machine Learning Research*, pages 1628–1638. PMLR, 2023.

[68] M. Papini, D. Binaghi, G. Canonaco, M. Pirotta, and M. Restelli. Stochastic variance-reduced policy gradient. In *International Conference on Machine Learning*, 2018.

[69] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[70] B. Poczos and J. Schneider. On the estimation of α-divergences. In *AISTATS*. PMLR, 2011.

[71] R. Ranganath, S. Gerrish, and D. M. Blei. Black box variational inference. In *AISTATS*, volume 33 of *JMLR Workshop and Conference Proceedings*, pages 814–822. JMLR.org, 2014.

[72] R. Ranganath, D. Tran, J. Altosaar, and D. M. Blei. Operator variational inference. In *NeurIPS*, pages 496–504, 2016.

[73] J. Rector-Brooks, K. Madan, M. Jain, M. Korablyov, C.-H. Liu, S. Chandar, N. Malkin, and Y. Bengio. Thompson sampling for improved exploration in gflownets, 2023.

[74] A. Rényi. On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. University of California Press, 1961.

[75] D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*. PMLR, 2015.

[76] B. Rhodes and M. U. Gutmann. Variational noise-contrastive estimation. In *AISTATS*, 2019.

[77] L. Richter, A. Boustati, N. Nüsken, F. J. R. Ruiz, and Ö. D. Akyildiz. Vargrad: A low-variance gradient estimator for variational inference. 2020.

[78] C. P. Robert et al. *The Bayesian choice: from decision-theoretic foundations to computational implementation*, volume 2. Springer, 2007.

[79] G. Roeder, Y. Wu, and D. Duvenaud. Sticking the landing: simple, lower-variance gradient estimators for variational inference. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6928–6937, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[80] T. G. J. Rudner, V. Pong, R. McAllister, Y. Gal, and S. Levine. Outcome-driven reinforcement learning via variational inference. In *NeurIPS*, pages 13045–13058, 2021.

[81] F. Ruiz and M. Titsias. A contrastive divergence for combining variational inference and mcmc. In *International Conference on Machine Learning*, 2019.

[82] T. Salimans and D. A. Knowles. On using control variates with stochastic approximation for variational bayes and its connection to stochastic linear regression, 2014.

[83] M. J. Schervish. *Theory of statistics*. Springer Science & Business Media, 2012.

[84] M. W. Shen, E. Bengio, E. Hajiramezanali, A. Loukas, K. Cho, and T. Biancalani. Towards understanding and improving gflownet training. In *International Conference on Machine Learning*, 2023.

[85] J. Shi, Y. Zhou, J. Hwang, M. Titsias, and L. Mackey. Gradient estimation with discrete stein operators. *NeurIPS*, 35, 2022.

[86] Y. Sun, L. Dong, S. Huang, S. Ma, Y. Xia, J. Xue, J. Wang, and F. Wei. Retentive network: A successor to transformer for large language models, 2023.

[87] D. Tiapkin, N. Morozov, A. Naumov, and D. Vetrov. Generative flow networks as entropy-regularized rl, 2024.

[88] E. Todorov. General duality between optimal control and estimation. In *CDC*, pages 4286–4292. IEEE, 2008.

[89] M. Toussaint and A. J. Storkey. Probabilistic inference for solving discrete and continuous state markov decision processes. In *ICML*, volume 148 of *ACM International Conference Proceeding Series*, pages 945–952. ACM, 2006.

[90] M.-N. Tran, M. K. Pitt, and R. Kohn. Adaptive metropolis–hastings sampling using reversible dependent mixture proposals. *Statistics and Computing*, 2016.

[91] C. Tsallis. Possible generalization of boltzmann-gibbs statistics. *Journal of statistical physics*, 52:479–487, 1988.

[92] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, 2008.

[93] X. Wang, T. Geffner, and J. Domke. Joint control variate for faster black-box variational inference. In *AISTATS*, volume 238 of *Proceedings of Machine Learning Research*, pages 1639–1647. PMLR, 2024.

[94] L. Weaver and N. Tao. The optimal reward baseline for gradient-based reinforcement learning. *arXiv preprint arXiv:1301.2315*, 2013.

[95] V. D. Wild, R. Hu, and D. Sejdinovic. Generalized variational inference in function spaces: Gaussian measures meet bayesian deep learning. In *NeurIPS*, 2022.

[96] D. Williams. *Probability with Martingales.* Cambridge University Press, 1991.

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

[98] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[99] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? *International Conference on Learning Representations (ICLR)*, 2019.

[100] P. Xu, F. Gao, and Q. Gu. An improved convergence analysis of stochastic variance-reduced policy gradient. In *UAI*, 2020.

[101] Z. Yang. *Molecular Evolution: A Statistical Approach*. Oxford University PressOxford, May 2014. ISBN 9780191782251. doi: 10.1093/acprof:oso/9780199602605.001.0001.

 https://doi.org/10.52202/079017-2416

[102] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Bethe free energy, kikuchi approximations, and belief propagation algorithms. *Advances in neural information processing systems*, 13(24), 2001.

[103] M. Yin and M. Zhou. Semi-implicit variational inference. In *International conference on machine learning*. PMLR, 2018.

[104] L. Yu and C. Zhang. Semi-implicit variational inference via score matching. *arXiv preprint arXiv:2308.10014*, 2023.

[105] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *NeurIPS*, 2017.

[106] C. Zhang, B. Shahbaba, and H. Zhao. Hamiltonian monte carlo acceleration using surrogate functions with random bases. *Statistics and computing*, 2017.

[107] D. Zhang, R. T. Chen, N. Malkin, and Y. Bengio. Unifying generative models with gflownets and beyond. *ICML Beyond Bayes workshop*, 2022.

[108] D. Zhang, N. Malkin, Z. Liu, A. Volokhova, A. Courville, and Y. Bengio. Generative flow networks for discrete probabilistic modeling. In *International Conference on Machine Learning (ICML)*, 2022.

[109] D. Zhang, H. Dai, N. Malkin, A. Courville, Y. Bengio, and L. Pan. Let the flows tell: Solving graph combinatorial optimization problems with gflownets. In *NeurIPS (NeurIPS)*, 2023.

[110] D. Zhang, R. T. Q. Chen, C.-H. Liu, A. Courville, and Y. Bengio. Diffusion generative flow samplers: Improving learning signals through partial trajectory optimization. In *The Twelfth International Conference on Learning Representations*, 2024.

[111] M. Y. Zhou, Z. Yan, E. Layne, N. Malkin, D. Zhang, M. Jain, M. Blanchette, and Y. Bengio. PhyloGFN: Phylogenetic inference with generative flow networks. In *The Twelfth International Conference on Learning Representations*, 2024.

[112] H. Zimmermann, F. Lindsten, J. van de Meent, and C. A. Naesseth. A variational perspective on generative flow networks. *Trans. Mach. Learn. Res.*, 2023, 2023.

A Related works

Generative Flow Networks. GFlowNets [3, 4] were initially proposed as a reinforcement learning algorithm for finding diverse high-valued states in a discrete environment by sampling from a distribution induced by a reward function. Shortly after, they were extended to sample from complex distributions in arbitrary topological spaces [48]. Remarkably, this family of models was successfully applied to problems as diverse as structure learning and causal discovery [13, 15, 16], discrete probabilistic modeling and graphical models [24, 29, 107, 108], combinatorial optimization and stochastic control [109, 110], drug discovery [3, 33, 62], design of biological sequences [32], natural language processing [30], and aerial scene classification [23]. Concomitantly to these advances, there is a growing literature concerned with the development of more efficient training algorithms for GFlowNets [4, 40, 55, 84] — primarily drawing inspiration from existing techniques in the reinforcement learning literature [60, 65, 66, 87]. In the same spirit, Tiapkin et al. [87] showed it is possible to frame GFlowNets as an entropy-regularized reinforcement learning. In a study closely related to ours, Malkin et al. [56] proved the equivalence between GFlowNets and hierarchical variational inference (HVI) for discrete distributions; however, when training GFlowNets using divergence-based methods from the VI literature, the authors found no improvement relatively to the traditional flow-matching objectives. Thus, extending beyond discrete distributions, this work provides a definitive analysis of training GFlowNets by directly optimizing a set of divergences typically employed in variational inference training, given a clear context and conditions for effective use of divergence objectives for efficient learning procedures applied on GFlowNets models.

Reinforcement Learning as Inference. Reinforcement Learning (RL) has been studied as a form of probabilistic inference extensively, generating relevant insights in the literature, and alternatively referred to as *control as inference*. Todorov [88] demonstrates a duality between estimation and optimal control, establishing conditions where estimation algorithms could applied for control problems. Kappen et al. [38] demonstrated that optimal control problems could be framed as inference problems in graphical models, providing a unified perspective for solving control tasks. Levine [50] presents a complete and modern RL formulation, linking with VI in particular. Rudner et al. [80] integrates even further RL with VI methods, demonstrating the conceptual and algorithmic gains of leveraging outcome-driven RL with variational inference to optimize policy distributions. Developing further, Toussaint and Storkey [89] applies approximate probabilistic inference methods to solve Markov Decision Processes (MDPs) with discrete and continuous states. The approach also aligns with model-based RL techniques, such as *PILCO*, which utilizes probabilistic models to enhance data efficiency in policy search [14]. Recent work by Deleu et al. [17] positions discrete probabilistic inference as a control problem in multi-path environments, highlighting the synergy between control theory and probabilistic modeling in the context of GFlowNets. This body of works relates to the approach presented in this paper, comparing optimization of trajectory balance and flow-matching losses related to sequential decisions modeled by the GFlowNet with f-divergence measures minimization procedures – related to approximated variational inference and generalized posterior inference [36, 45, 51, 92, 95].

Divergence measures and gradient reduction for VI. Approximate inference via variational inference (VI) methods [6, 7, 36, 92] initially relied on message passing and coordinate ascent methods to minimize the KL divergence of an unnormalized distribution and a proposal in a parameterized tractable family of distributions. Despite the initial generality of the optimization perspective, the concrete implementation of algorithms often requires specialized update equations and learning objectives for specific classes of models. On the other hand, the development of algorithms and software for automatic differentiation [2] and stochastic gradient estimators [59] unlocked the potential application of generic gradient-based optimization algorithms in inference and learning tasks for a comprehensive class of models. Seminal works such as Black-Box VI (BBVI) [71], using the REINFORCE/score function estimator, and Automatic Differentiation VI (ADVI) [46], using reparameterization and change-of-variables, demonstrated practical algorithms for Bayesian inference in generic models, including models combining classical statistical modeling with neural networks. Overall, Mohamed et al. [59] explain the development of the main gradient estimators: the score function [11, 71, 97, 103], and the pathwise gradient estimator, also known as the parametrization trick [42, 43, 75]. The vanilla REINFORCE/score function estimator has notoriously high variance [11, 71, 77, 97], which prompted a body of work exploring variance reduction techniques. In the original BBVI proposal, Ranganath et al. [71] explored Rao-Blackwellization, combining iterated conditional expectations and control variates, using the score function estimator (given its zero

 https://doi.org/10.52202/079017-2416

expectation) as a control variate. Subsequent works have continued to refine these techniques; Liu et al. [53] uses Rao-Blackwellized stochastic gradients for discrete distributions, while Kim et al. [39] and Wang et al. [93] explored joint control variates and provable linear convergence in BBVI. Additionally, Domke [21] and Domke [20] provided smoothness and gradient variance guarantees, further enhancing the robustness of score function estimator for VI methods. Our work demonstrates that effective variance reduction techniques applied to a f-divergence minimization training can significantly enhance the convergence speed and stability of the procedure. In theory and practice, we observed high compatibility between our results of variance-reduced f-divergence GFlowNets training and the body of work of variance-reduced score-function estimators for VI. Furthermore, by showing that these techniques apply to a broad class of models and optimization objectives, including continuous and mixed structured supports, we move GFlowNets' f-divergence minimization training closer to recent notions of generalized Bayesian inference and generalized VI[45] and variational inference in function spaces [95] – with the common thread of casting posterior inference as an optimization problem guided by some divergence measure. This generalization can enable applications of GFlowNets to a diverse range of machine learning tasks, enhancing their versatility and practical relevance.

B Detailed description of the generative tasks

Set generation [3, 34, 65, 66]**.** $\mathcal{S}$ contains the sets of size up to N with elements extracted from a fixed deposit $\mathcal{D}$ of size $D \geq N$ and $s_o = \emptyset$. For $s \in \mathcal{S}$ with $|s| < N$, $\kappa_f(s, \cdot)$ is a counting measure supported at (the σ-algebra induced by) $\{s \cup \{d\} \colon d \in \mathcal{D} \setminus s\}$; for $|s| = N$, $\kappa_f(s, \cdot) = \delta_{s_f}$. Then, $\mathcal{X} = \{s \in \mathcal{S} \colon |s| = N\}$. Similarly, $\kappa_b(s, \cdot)$'s support is $\{s \setminus \{d\} \colon d \in \mathcal{D}\}$ for $s \neq s_o$. We define the unnormalized target's density by $\log r(x) = \sum_{d \in x} f(d)$ for a fixed function $f \colon \mathcal{D} \to \mathbb{R}$. We parameterize $p_F(s, \cdot)$ as a deep set [105] and fix $p_B(s, \cdot)$ as a uniform density for $s \in \mathcal{S}$.

Autoregressive sequence generation [32, 55]**.** A *sequence* s in $\mathcal{D}^n$, for any $K > n$, is bijectively associated to an element of $\mathcal{D} \times [K]$ by $s \mapsto \{(s_m, m) \colon 1 \leq m \leq n\} \cup \{(\perp, m) \colon K \geq m > n\}$; $\perp$ is a token denoting the sequence's end. In this context, we let $\mathcal{S} \subset \mathcal{P}(\mathcal{D} \times [N+1])$ be the set of sequences of size up to N, i.e., if $s \in \mathcal{S}$ and $(\perp, n+1) \in s$, then $(d, m) \in s$ iff $d = \perp$ for $n < m \leq N+1$ and there is $v \in (\mathcal{D} \cup \{\perp\})^n$ such that $(v_m, m) \in s$ for $m \leq n$; the initial state is $s_o = \emptyset$. For conciseness, we write $d \notin s$, meaning that $(d, i) \notin s$ for every i. Next, $\kappa_f(s, \cdot)$ is the counting measure supported at $\{s \cup \{(d, |s|+1)\} \colon d \in \mathcal{D} \cup \{\perp\}\}$ if $|s| < N$ and $\perp \notin s$; at $\{s \cup \{(\perp, N+1)\}\}$ if $|s| = N$; and at $\{s_f\}$ otherwise. Thus, $\mathcal{X} = \{s \in \mathcal{S} \colon \perp \in s\}$. Also, $k_b(s, \cdot)$ is supported at $\{s \setminus \{(d, |s|)\} \colon d \in \mathcal{D}\}$, which has a single element corresponding to the removal of the element of s of the largest index. On the other hand, the target's density is $\log r(x) = \sum_{(d,i) \in x \colon d \neq \perp} f(d) g(i)$ for functions $f, g \colon \mathcal{D} \to \mathbb{R}$. We employ an MLP to parameterize $p_F(s, \cdot)$.

Bayesian phylogenetic inference (BPI) [111]**.** A (rooted) *phylogeny* is a complete binary tree with labeled leaves and weighted edges; each leaf corresponds to a biological species, and the edges' weights are a measurement of evolutionary time. Formally, we let $\mathcal{B}$ be the set of observed biological species and $\mathcal{V}_\mathcal{B}$ be the set of $|\mathcal{B}| + 1$ unobserved species. Next, we represent a phylogeny over $\mathcal{B}$ as a weighted directed tree $G_\mathcal{B} = (\mathcal{B} \cup \mathcal{V}_\mathcal{B}, E_\mathcal{B}, \omega_\mathcal{B})$ with edges $E_\mathcal{B}$ featured with a weight assignment $\omega_\mathcal{B}$; we denote its root by $r(G_\mathcal{B})$. Under these conditions, we define $\mathcal{S} = \left\{\bigcup_{k=1}^K G_{\mathcal{F}_k} \colon \bigsqcup_k \mathcal{F}_k = \mathcal{B} \wedge \mathcal{G}_{\mathcal{F}_k} \text{ is a tree}\right\}$ as the set of forests built upon phylogenetic trees over disjoint subsets of $\mathcal{B}$; $\bigsqcup$ represents a disjoint union of non-empty sets and $s_o = \bigcup_{b \in \mathcal{B}} G_{\{b\}}$ is the forest composed of singleton trees $G_{\{b\}}$. Also, we define the *amalgamation* of phylogenies $G_{\mathcal{F}_k}$ and $G_{\mathcal{F}_j}$, $\mathcal{A}(G_{\mathcal{F}_k}, G_{\mathcal{F}_j})$, as the tree obtained by joining their roots $r(G_{\mathcal{F}_k})$ and $r(G_{\mathcal{F}_j})$ to a new node $r(G_{\mathcal{F}_k} \cup G_{\mathcal{F}_j})$, with a corresponding extension of the edges' weights. In contrast, the *dissolution* of a tree $G_\mathcal{F}$, $\mathcal{R}(G_\mathcal{F})$, returns the union of the two subtrees obtained by removing $r(G_\mathcal{F})$ from $G_\mathcal{F}$. Then, $\kappa_f(s, \cdot)$ is the counting measure supported at $\left\{\bigcup_{k=1, k \neq i, j}^K G_{\mathcal{F}_k} \cup \mathcal{A}(G_{\mathcal{F}_i}, G_{\mathcal{F}_j}) \colon (i, j) \in [K]^2,\, i \neq j\right\}$ with $s = \bigcup_{k=1}^K G_{\mathcal{F}_k}$ and $K \geq 2$; if $s = \mathcal{G}_\mathcal{B}$, $\kappa_f(s, \cdot) = \delta_{s_f}$. Hence, $\mathcal{X}$ is the set of phylogenies over $\mathcal{B}$. Likewise, $\kappa_b(s, \cdot)$'s support is $\left\{\bigcup_{k=1, k \neq j}^K G_{\mathcal{F}_k} \cup \mathcal{R}(G_{\mathcal{F}_i}) \colon i \in [K] \wedge r(G_{\mathcal{F}_i}) \notin \mathcal{B}\right\}$ for $s = \bigcup_{k=1}^K G_{\mathcal{F}_k}$ and $K \leq |\mathcal{B}|$. Finally, the unnormalized target is the posterior distribution defined by JC69's mutation model [37] given a data set of genome sequences of the species in $\mathcal{B}$. More specifically, we let the prior be a uniform distribution and compute the model-induced likelihood function by the efficient Felsenstein's algorithm [25]. We assume the edges' weights are constant. See [101] for further details. We parameterize $p_F(s, \cdot)$ with GIN [99] and fix $p_B(s, \cdot)$ as an uniform distribution.

Mixture of Gaussians [48, 110]**.** The training of GFlowNets in continuous spaces is challenging, and the problem of designing highly expressive models in this setting is still unaddressed [16, 48]. However, as we show in Section 5.2, divergence-based measures seem to be very effective learning objectives for autoregressive sampling of a sparse mixture of Gaussians. For a d-dimensional Gaussian distribution, $\mathcal{S} = \{\{(0,0), (x_i, i) \colon 1 \leq i \leq n\}, : n \leq d,\, x \in \mathbb{R}^n\} \subset \{(0,0)\} \cup \mathcal{P}(\mathbb{R} \times [d])$ and $s_o = (0,0)$; note $\mathcal{S}$ is isomorphic to $\mathbb{R}^d$. Also, for $s = \{(x_i, i)\}_{i=1}^n$, $\kappa_f(s, \cdot)$ is Lebesgue's measure at $\{s \cup (x, n+1) \colon x \in \mathbb{R}\}$ if $n < d$ and $\kappa_f(s, \cdot) = \delta_{s_f}$ otherwise. In particular, $\mathcal{X} = \{s \in \mathcal{S} \colon \max_{(x,i) \in s} i = d\}$. Moreover, $\kappa_b(s, \cdot)$ is a measure on $\{s \setminus (x, |s|) \colon x \in \mathbb{R}\}$, which is isomorphic to $\mathbb{R}$, which is a singleton. We define the target's density with a homogeneous mixture of Gaussian distributions, $\frac{1}{K} \sum_{i=1}^K \mathcal{N}(\mu_i, \sigma^2 I)$ with $\mu_i \in \mathbb{R}^d$. We similarly define $P_F(s, \cdot)$ as a mixture of one-dimensional Gaussians with mean and variance learned via an MLP [48].

https://doi.org/10.52202/079017-2416

Banana distribution. [57, 76] We consider sampling from the banana distribution, defined by

$$\mathcal{N}\left(\begin{bmatrix} x_1 \\ x_2 + x_1^2 + 1 \end{bmatrix} \middle| \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix}\right). \tag{8}$$

Given its geometry and shape, this distribution is a common baseline in the approximate Bayesian inference literature [90, 104, 106]. This task is identical to sampling from a mixture of Gaussian distributions, except for the different target density specified by the model in Equation (8). Also, we rely on the implementation Hamiltonian Monte Carlo (HMC) [5, 61] provided by Stan [12] to obtain accurate samples from (8).

A similar description may be utilized for the hypergrid and structure learning domains.

C Proofs

We will consider the measurable space of *trajectories* $(\mathcal{P}_{\mathcal{S}}, \Sigma_P)$, with $\mathcal{P}_{\mathcal{S}} = \{(s, s_1, \ldots, s_n, s_f) \in \mathcal{S}^{n+1} \times \{s_f\} \colon 0 \leq n \leq N-1\}$ and Σ_P as the σ-algebra generated by $\bigcup_{n=1}^{N+1} \Sigma^{\otimes n}$. For notational convenience, we use the same letters for representing the measures and kernels of $(\mathcal{S}, \Sigma)$ and their natural product counterparts in $(\mathcal{P}_{\mathcal{S}}, \Sigma_P)$, which exist by Carathéodory extension's theorem [96]; for example, $\nu(B) = \nu^{\otimes n}(B)$ for $B = (B_1, \ldots, B_n) \in \Sigma^{\otimes n}$ and $p_{F_\theta}(\tau|s_o;\theta)$ is the density of $P_F^{\otimes n+1}(s_o, \cdot)$ for $\tau = (s_o, s_1, \ldots, s_n, s_f)$ relatively to $\mu^{\otimes n}$. In this case, we will write τ for a generic element of $\mathcal{P}_{\mathcal{S}}$ and x for its terminal state (which is unique by Definition 1).

C.1 Proof of Proposition 1

We will show that the gradient of the expected on-policy TB loss matches the gradient of the KL divergence between the forward and backward policies. Firstly, note that

$$\begin{aligned}
\nabla_\theta \mathcal{D}_{KL}[P_F||P_B] &= \nabla_\theta \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[\log \frac{p_F(\tau|s_o;\theta)}{p_B(\tau)}\right] \\
&= \nabla_\theta \int_\tau \log \frac{p_F(\tau|s_o;\theta)}{p_B(\tau)} \mathrm{d}P_F(s_o, \mathrm{d}\tau) \\
&= \nabla_\theta \int_\tau \log \frac{p_F(\tau|s_o;\theta)}{p_B(\tau)} p_F(\tau|s_o;\theta) \mathrm{d}\kappa_f(s_o, \mathrm{d}\tau) \\
&= \int_\tau \nabla_\theta \log \frac{p_F(\tau|s_o;\theta)}{p_B(\tau)} P_F(s_o, \mathrm{d}\tau) \\
&+ \int_\tau \log \frac{p_F(\tau|s_o;\theta)}{p_B(\tau)} \nabla_\theta p_F(\tau|s_o;\theta) \mathrm{d}\kappa_f(s_o, \mathrm{d}\tau)
\end{aligned}$$

by Leibniz's rule for integrals and the product rule for derivatives. Then, since $\nabla_\theta f(\theta) = f(\theta)\nabla \log f(\theta)$ for any differentiable function $f \colon \theta \mapsto f(\theta)$,

$$\begin{aligned}
&\nabla_\theta \mathcal{D}_{KL}\left[P_F||P_B\right] \\
&= \mathop{\mathbb{E}}_{\tau\sim P_F(s_o,\cdot)}\left[\nabla_\theta \log p_F(\tau|s_o) + \log \frac{p_F(\tau|s_o)}{p_B(\tau)} \nabla_\theta \log p_F(\tau|s_o)\right] \\
&= \mathop{\mathbb{E}}_{\tau\sim P_F(s_o,\cdot)}\left[\log \frac{p_F(\tau|s_o)}{p_B(\tau)} \nabla_\theta \log p_F(\tau|s_o)\right];
\end{aligned} \tag{9}$$

we omitted the dependency of P_F (and of p_F thereof) on the parameters θ for conciseness. On the other hand,

$$\nabla_\theta \mathcal{L}_{TB}(\tau;\theta) = 2\left(\log \frac{p_F(\tau|s_o;\theta)}{p_B(\tau)}\right) \nabla_\theta \log p_F(\tau) \tag{10}$$

by the chain rule for derivatives. Thus,

$$\mathbb{E}_{\tau\sim P_F(s_o,\cdot)} \nabla_\theta \mathcal{L}_{TB}(\tau;\theta) = 2\nabla_\theta \mathcal{D}_{KL}[P_F||P_B], \tag{11}$$

ensuring that the equivalence between $\mathcal{L}_{TB}$ and $\mathcal{D}_{KL}$ in terms of expected gradients holds in a context broader than that of finitely supported distributions [56].

C.2 Proof of Lemma 1

Henceforth, we will recurrently refer to the score estimator for gradients of expectations [97], namely,

$$\nabla_\theta \mathop{\mathbb{E}}_{\tau\sim P_F(s_o,\cdot)}\left[f_\theta(\tau)\right] = \mathop{\mathbb{E}}_{\tau\sim P_F(s_o,\cdot)}\left[\nabla_\theta f_\theta(\tau) + f_\theta(\tau)\nabla_\theta \log p_F(\tau|s_o;\theta)\right], \tag{12}$$

which can be derived using the arguments of the preceding section. In this context, the Renyi-α's divergence satisfies

$$\nabla_\theta R_\alpha(P_F||P_B) = \frac{\nabla_\theta \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}[g(\tau,\theta)]}{(\alpha-1)\mathbb{E}_{\tau\sim P_F(s_o,\cdot)} g(\tau,\theta)},$$

with $g(\tau;\theta) = \left({}^{p_B(\tau|x)r(x)}/_{p_F(\tau|s_o;\theta)}\right)^{1-\alpha}$ and $\alpha \neq 1$; similarly, the Tsallis-α's divergence abides by

$$\nabla_\theta T_\alpha(P_F||P_B) = \frac{1}{(\alpha-1)} \nabla_\theta \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}[g(\tau,\theta)]. \tag{13}$$

https://doi.org/10.52202/079017-2416

The statement then follows by substituting $\nabla_\theta \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}[g(\tau,\theta)]$ with the corresponding score estimator given by Equation (12).

C.3 Proof of Lemma 2

Forward KL divergence. The gradient of $\mathcal{D}_{KL}[P_B||P_F]$ is straightforwardly obtained through the application of Leibniz's rule for integrals,

$$\nabla_\theta \mathcal{D}_{KL}[P_B||P_F] = -\mathbb{E}_{\tau\sim P_B(s_f,\cdot)}\left[\nabla_\theta \log p_F(\tau|s_o;\theta)\right],$$

since the averaging distribution P_B do not depend on the varying parameters θ. However, as we compute Monte Carlo averages over samples of P_F, we apply an importance reweighting scheme [63, Chapter 9] to the previous expectation to infer that, up to a positive multiplicative constant,

$$\nabla_\theta \mathcal{D}_{KL}[P_B||P_F] \overset{C}{=} -\mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[\frac{p_B(\tau|x)r(x)}{p_F(\tau|s_o;\theta)}\nabla_\theta \log p_F(\tau|s_o;\theta)\right],$$

with $\overset{C}{=}$ denoting equality up to a positive multiplicative constant. We emphasize that most modern stochastic gradient methods for optimization, such as Adam [41] and RMSProp [28], remain unchanged when we multiply the estimated gradients by a fixed quantity; thus, we may harmlessly compute gradients up to multiplicative constants.

Reverse KL divergence. We verified in Equation (9) that

$$\nabla_\theta \mathcal{D}_{KL}[P_F||P_B] = \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[\log\frac{p_F(\tau|s_o)}{p_B(\tau)}\nabla_\theta s_\theta(\tau)\right].$$

Since $p_B(\tau) = p_B(\tau|x)^{r(x)}/Z$ and $\mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\nabla_\theta s_\theta(\tau) = 0$, the quantity $\nabla_\theta \mathcal{D}_{KL}[P_F||P_B]$ may be rewritten as

$$\begin{aligned}\nabla_\theta \mathcal{D}_{KL}[P_F||P_B] &= \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[\log\frac{p_F(\tau|s_o)}{p_B(\tau)}\nabla_\theta s_\theta(\tau)\right] \\ &\quad + \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[(\log Z)\nabla_\theta s_\theta(\tau)\right] \\ &= \mathbb{E}_{\tau\sim P_F(s_o,\cdot)}\left[\log\frac{p_F(\tau|s_o)}{p_B(\tau|x)r(x)}\nabla_\theta s_\theta(\tau)\right],\end{aligned}$$

in which x is the terminal state corresponding to the trajectory τ. Thus proving the statement in Lemma 2.

C.4 Proof of Proposition 2

We will derive an expression for the optimal baseline of a vector-valued control variate. For this, let f be the averaged function and $g\colon \tau \mapsto g(\tau)$ be the control variate. Assume, without loss of generality, that $\mathbb{E}_\pi[g] = 0$ for the averaging distribution π over the space of trajectories. In this case, the optimal baseline for the control variate $a^\star$ is found by

$$a^\star = \arg\min_{a\in\mathbb{R}} \operatorname{Tr}\left(\operatorname{Cov}_{\tau\sim\pi}[f(\tau) - a\cdot g(\tau)]\right). \tag{14}$$

Thus,

$$a^\star = \arg\min_{a\in\mathbb{R}} \operatorname{Tr}\Big(-2a\cdot\operatorname{Cov}_\pi[f(\tau),g(\tau)] + a^2\operatorname{Cov}_\pi(g(\tau))\Big),$$

which is a convex optimization problem solved by

$$\begin{aligned}a^\star &= \frac{\operatorname{Tr}\left(\operatorname{Cov}_\pi[f(\tau),g(\tau)]\right)}{\operatorname{Tr}\left(\operatorname{Cov}_\pi[g(\tau)]\right)} \\ &= \frac{\operatorname{Tr}\,\mathbb{E}_\pi[(f-\mathbb{E}_\pi[f(\tau)])g(\tau)^T]}{\operatorname{Tr}\,\mathbb{E}_\pi[g(\tau)g(\tau)^T]} \\ &= \frac{\mathbb{E}_\pi[\operatorname{Tr}\left(f-\mathbb{E}_\pi[f(\tau)]\right)^T g(\tau)]}{\mathbb{E}_\pi[\operatorname{Tr}\,g(\tau)^T g(\tau)]} \\ &= \frac{\mathbb{E}_\pi[(f-\mathbb{E}_\pi[f(\tau)])^T g(\tau)]}{\mathbb{E}_\pi[g(\tau)^T g(\tau)]},\end{aligned} \tag{15}$$

in which we used the circular property of the trace. This equation exactly matches the result in Proposition 2. In practice, we use $g(\tau) = \nabla_\theta \log p_F(\tau)$ for both the reverse KL- and α-divergences, rendering a baseline $a^\star$ that depends non-linearly on the sample gradients and is hence difficult to compute in a GPU-powered autodiff framework efficiently. We thus use Equation (6) to estimate $a^\star$.

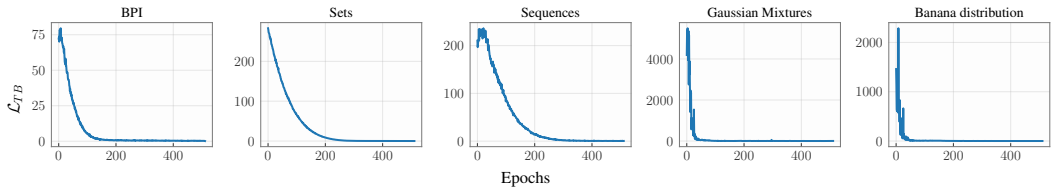

Figure 7: **Learning curves for a GFlowNet trained by minimizing the TB loss.** The curves' smoothness highlights the low variance of the optimization steps incurred by the stochastic gradients of $\mathcal{L}_{TB}$, which do *not* use a score function estimator.

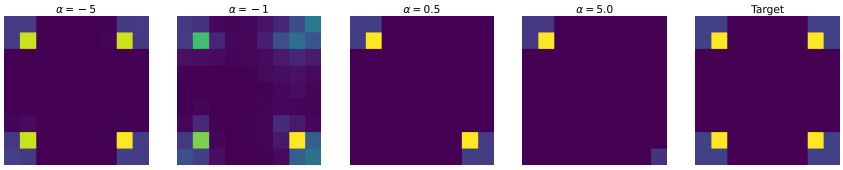

Figure 9: **Additional illustration of the effect of** α when learning GFlowNets by minimizing the Renyi-α divergence in the hypergrid environment. For such a sparse target distribution, a large and negative value of α (left) is beneficial to ensure that all modes are appropriately covered by the learned distribution. In contrast, the mode-seeking behavior induced by a large value of α entails the collapse of the model in a single mode (right).

D Additional experiments

Gradient variance for flow-network-based objectives. Figure 7 shows the learning curve for the TB loss in each of the generative tasks. Notoriously, it underlines the low variance of the optimization steps — which, contrarily to their divergence-based counterparts, do not rely on a score function estimator — and suggests that the design of control variates for estimating the gradients of these objectives would not be significantly helpful. Also, the gradient of $\mathcal{L}_{TB}$ depends non-linearly on the score function $\log p_F$ and, consequently, it is unclear how to implement computationally efficient variance reduction techniques in this case.

Forward KL for sequence generation. Figure 3 shows that alternative approaches in terms of convergence speed outperformed a GFlowNet trained to minimize the forward KL. One possible cause of this underperformance is the high variance induced by the underlying importance sampling estimator. To verify this, we re-run the corresponding experiments, increasing the size of the batch of trajectories for the forward KL estimator to 1024. Figure 8 presents the experiment's results, with an increased batch size corresponding to an estimator of smaller variance that accelerates the GFlowNet's training convergence. More broadly, this suggests that the design of GFlowNet-specific variance reduction techniques, which we leave to future endeavors, may further improve this family of models.

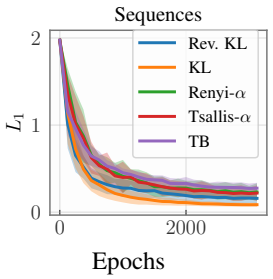

Figure 8: Results for sequence generation with larger batches.

Influence of α on the learning of GFlowNets. As mentioned earlier, divergence-based measures fall short compared to their balance-based counterparts for the hypergrid navigation task. For this extremely sparse problem, the benefits from off-policy exploration during training seem to supersede the statistical efficiency enacted by the minimization of divergences, which fail to properly cover the high-probability regions of the target distribution. In this scenario, Figure 9 suggests that this issue can be mildly circumvented by choosing a sufficiently negative α for the Renyi-α divergences, thereby imposing a mass-covering behavior to the learned model. However, these results should be substantiated by further experimental analysis to be conclusive. Currently, we would suggest a practitioner to stick to the balance-based objectives when dealing with very sparse target distributions.

https://doi.org/10.52202/079017-2416

E Experimental details

The following paragraph provides further implementation details. Regarding open access to the code, we will make the code public upon acceptance.

Shared configurations. For every generative task, we used the Adam optimizer [41] to carry out the stochastic optimization, employing a learning rate of 10^{-1} for $\log Z_\theta$ when minimizing $\mathcal{L}_{TB}$ and 10^{-3} for the remaining parameters, following previous works [48, 55, 56, 66]. We polynomially annealed the learning rate towards 0 along training, similarly to [86]. Also, we use LeakyReLU [98] as the non-linear activation function of all implemented neural networks.

Set generation. We implement an MLP of 2 64-dimensional layers to parameterize the policy's logits $\log p_F(s, \cdot)$. We train the model for 512 epochs with a batch of 128 trajectories for estimating the gradients. Also, we let $D = 32$ and $N = 16$ be the source's and set's sizes, respectively.

Autoregressive sequence generation. We parameterize the logits of the forward policy with a MLP of 2 64-dimensional layers; we pad the sequences to account for their variable sizes. We respectively consider $D = 8$ and $N = 6$ for the source's and sequence's sizes. To approximate the gradients, we rely on a batch of 128 sequences.

Bayesian phylogenetic inference. We parameterize the logits of the forward policy with a 2-layer GIN [99] with a 64-dimensional latent embedding, which is linearly projected to $\log p_F$. Moreover, we simulated the JC69 model [37] to obtain 25-sized sequences of nucleotides for each of the 7 observed species, setting $\lambda = 0.3$ for the instantaneous mutation rate; see [101] for an introduction to computational phylogenetics and molecular evolution. To estimate the gradients, we relied on batches of 64 trajectories.

Hypergrid navigation. We consider a $H = 12$ for Figure 3 and Figure 4 and $H = 9$ for Figure 9. In both cases, $d = 2$ and $R_o = 10^{-3}$; see [55, Section 5.1]. To parameterize the policy, we used a 2-layer 256-dimensional MLP with ReLU activations. We trained the models for 10000 epochs.

Bayesian structure learning. We simulated a 100-sized 5-variable data set $\mathbf{X}$ from a randomly parameterized linear Gaussian structural model based on a fixed Bayesian network. We implemented a 2-layer 256-dimensional MLP with ReLU activations for the policy network, which received the flattened adjacency matrix of the current state as input. Training lasted for 4000 epochs.

Mixture of Gaussian distributions. We consider a mixture of 9 2-dimensional Gaussian distributions centered at $\mu_{ij} = (i, j)$ for $0 \leq i, j \leq 2$, each of which having an isotropic variance of 10^{-1}; see Figure 1. We use an MLP of 2 64-dimensional layers to parameterize the forward policy.

Banana-shaped distribution. The model is specified by Equation (8). We also consider an MLP of 2 64-dimensional layers to parameterize the forward policy.

NeurIPS Paper Checklist

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: the proofs for the main claims in Proposition 1, Lemma 1, Proposition 2, Lemma 2 are presented in Appendix C, and practical training of GFN empirical validation in Section 5.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The middle paragraph in Section 6 discusses limitations, including the potential impact of different α values and the need for further exploration in this area.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

https://doi.org/10.52202/079017-2416

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The statements of the propositions and lemma include the assumptions, as well as in the notations and definitions presented in Section 4, and proofs provided in Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Details are provided in Section 5, Appendix B and Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Provided in a zip file during the review period, with the plan for public release once the paper is made public.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

https://doi.org/10.52202/079017-2416

Justification: Section 5 in the main text presents information about the experiments with full details provided in Appendix B and Appendix E, and supplemented zip-file with the code.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Plots count on error bars and tables count on standard deviation.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: First paragraph of Appendix E.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Our submission follow the NeurIPS ethical guidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We provide a perspective on broader impacts in Section 6, but do not foresee any direct negative societal impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

https://doi.org/10.52202/079017-2416

Justification: We do not foresee any direct risk stemming from our work.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [NA]

Justification: All code was made by the authors

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: No new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No experiments with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No experiments with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-2416