AVOID WHAT YOU KNOW: DIVERGENT TRAJECTORY BALANCE FOR GFLOWNETS

Pedro Dall'Antonia, Daniel Csillag & Diego Mesquita
School of Applied Mathmatics
Getulio Vargas Foundation

Tiago da Silva & Salem Lahlou
MBZUAI

ABSTRACT

Generative Flow Networks (GFlowNets) are a flexible family of amortized samplers trained to generate discrete and compositional objects with probability proportional to a reward function. However, learning efficiency is constrained by the model's ability to rapidly explore diverse high-probability regions during training. In this context, we propose *Adaptive Complementary Exploration* (ACE), a principled algorithm for the effective exploration of novel and high-probability regions when learning GFlowNets. To achieve this, ACE introduces an *exploration* GFlowNet explicitly trained to search for high-reward states in regions underexplored by the *canonical* GFlowNet, which learns to sample from the target distribution. Through extensive experiments, we show that ACE consistently and significantly improves upon prior work in terms of approximation accuracy to the target distribution and discovery rate of diverse high-reward states.

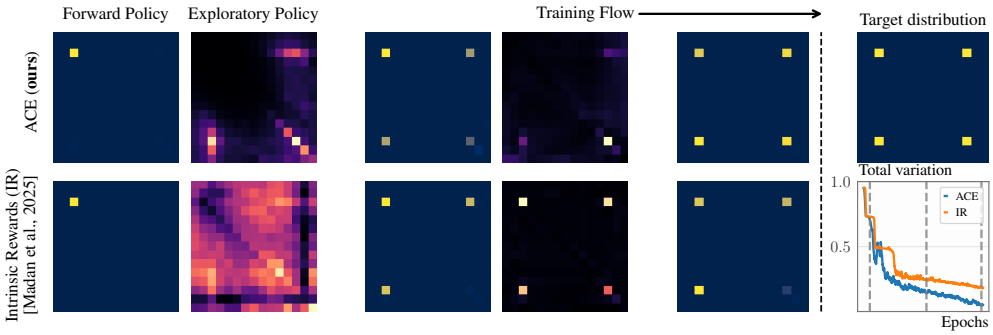


Figure 1: When learning the exploratory policy, competing methods may overemphasize well-learned states (bottom row). In contrast, ACE *avoids* sampling trajectories from over-explored regions of the state space by design (top row), which improves mode discovery and accelerates learning convergence to the target (rightmost panel). This figure shows the marginal distribution of forward and exploratory policies at different training points (marked as dashed vertical lines in the plot).

1 INTRODUCTION

Generative Flow Networks (GFlowNets; Bengio et al., 2021) are powerful reward-driven generative models designed to sample from distributions over compositional objects (e.g., graphs and sequences), with a range of applications in scientific discovery Wang et al. (2023), combinatorial optimization Zhang et al. (2023b;a), and approximate inference Malkin et al. (2023). Building on the compositional structure of the target distribution's support, GFlowNets create valid samples by starting from an initial state and iteratively drawing from a forward policy. Learning a GFlowNet then boils down to finding policies that satisfy a set of identities called *balance conditions*, which ensure sampling correctness.

To achieve this, we train GFlowNets by minimizing the logarithmic residuals of a balance condition over the state graph Malkin et al. (2022); Tiapkin et al. (2024). Nonetheless, since analytically sweeping through the entire state graph is intractable, we instead average over the residuals in a set of sampled trajectories. Conventionally, these trajectories are drawn from an ϵ-greedy *exploratory*

policy consisting of a mixture of the forward and an uniform policy at each time step. In theory, the uniform component provides full support to the sampling distribution, preventing mode collapse Bengio et al. (2023). In practice, however, uniform search might not be enough to warrant the exploration of multiple high-probability regions if the target reward is sparsely distributed (Malkin et al., 2023; Shen et al., 2023).

To improve exploration and provide better support coverage, recent works considered learning the exploratory policy alongside the GFlowNet. Inspired by curiosity-driven exploration, Kim et al. (2025b) recently suggested training such a policy to sample from a log-linear mixture of the GFlowNet's loss and the reward function. Concurrently, Madan et al. (2025) proposed instead targeting a combination of the original (extrinsic) reward and an intrinsic reward based on self-supervised random network distillation (RND). In both cases, exploration is guided by a GFlowNet steered by the loss function of an external deep neural network—either that of the underlying sampler Kim et al. (2025b); Malek et al. (2026) or the RND loss Madan et al. (2025). To distinguish the learned exploratory policy from the GFlowNet trained to sample from the target distribution, we will refer to the latter as the *canonical* GFlowNet.

From a fundamental viewpoint, the crux of designing an appropriate exploratory policy is ensuring it samples trajectories from high-reward but underexplored regions during training. In this work, we cast this problem as satisfying a new balance condition for exploration, which we call *divergent trajectory balance* (DTB). In a nutshell, DTB enforces standard trajectory balance on under-sampled regions while imposing zero probability to trajectories terminating in over-sampled states. As a consequence, any exploratory policy satisfying DTB concentrates its sampling on high-reward terminal states under-represented by the canonical GFlowNet.

The ensuing algorithm, called *Adaptive Complementary Exploration* (ACE), can be interpreted as a two-player game: the exploratory policy is continually adjusted to sample from under-sampled areas, and the GFlowNet being trained is updated using trajectories generated by the exploratory policy and its own policy. In doing so, we prevent the canonical GFlowNet from repeatedly visiting only a small number of high-reward subsets of the state space, while ignoring others, a behavior known to slow down training Vemgal et al. (2023); Atanackovic & Bengio (2024). We also characterize the system's equilibrium in Propositions 3.4 and 3.9.

When viewed through the lens of the flow network analogy, which inspired the development of GFlowNets Bengio et al. (2021), ACE implements a principled mechanism for transferring excess probability mass from over- to under-allocated nodes. We illustrate this in the diagram of Figure 1. Notably, our experiments on peptide discovery, bit sequence design, combinatorial optimization, and more, confirm ACE significantly speeds up training convergence and the discovery of diverse, high-reward states when compared against prior techniques for improved GFlowNet exploration.

2 PRELIMINARIES

Definitions. Our objective is to sample objects x from a discrete space $\mathcal{X}$ in proportion to a *reward* function $R : \mathcal{X} \to \mathbb{R}_+$. We say $\mathcal{X}$ is *compositional* if there is a directed acyclic graph $G = (\mathcal{S} \cup \mathcal{X}, \mathcal{E})$ over an extension $\mathcal{S} \cup \mathcal{X}$ of $\mathcal{X}$ with edges $\mathcal{E}$ having the following properties.

1. There exists a unique $s_o \in \mathcal{S}$ s.t. (i) s_o is connected to any $s \in \mathcal{S}$ via a directed path, denoted $s_o \rightsquigarrow s$, and (ii) s_o has no incoming edges. We call s_o the *initial state*.
2. There exists a unique $s_f \in \mathcal{S}$ s.t. (i) $s \to s_f \in \mathcal{E}$ if and only if $s \in \mathcal{X}$ and (ii) s_f has no outgoing edges. We refer s_f as the *final state* and to $\mathcal{X}$ as the set of *terminal states*.

Under these conditions, we call G a state graph and the GFlowNet its associated amortized sampler.

To start with, we recast the problem of directly sampling from R on $\mathcal{X}$ to that of learning an *amortized policy function* $p_F \colon \mathcal{S} \times (\mathcal{S} \cup \mathcal{X}) \to [0, 1]$ such that $p_F(s, \cdot)$ is a probability measure supported on the children of s in G. To achieve this goal, we parameterize $p_F(s, \cdot)$ as a softmax deep neural network trained to satisfy

$$p_\top(x) \coloneqq \sum_{\tau \colon s_o \rightsquigarrow x} \prod_{(s,s') \in \tau} p_F(s, s') \propto R(x), \tag{1}$$

in which the sum covers all trajectories from s_o to x in G. We refer to p_F as the *forward policy* and $p_\top(\cdot)$ as its induced marginal distribution over $\mathcal{X}$. For conciseness, we will often omit s_o and write $p_F(\tau) \coloneqq \prod_{(s,s')\in\tau} p_F(s, s')$ as the forward probability of a trajectory τ in G. As exact computation of $p_\top$ is intractable, we introduce a *backward* policy $p_B \colon (\mathcal{S} \cup \mathcal{X}) \times \mathcal{S} \to [0, 1]$ on the transposed state graph $G^\intercal$, and jointly search for p_F and p_B satisfying the *trajectory balance* (TB) condition for a learned constant Z,

$$Z \cdot p_F(\tau) = p_B(\tau|x) \cdot R(x), \tag{2}$$

in which $p_B(\tau|x) = \prod_{(s,s')\in\tau} p_B(s', s)$ is the backward probability of τ. As shown by Malkin et al. (2022); Madan et al. (2022), this can be achieved by solving the following stochastic program over Z and policies p_F and p_B,

$$\min_{Z,p_F,p_B} \mathbb{E}_{\tau\sim p_E}\left[\left(\log \frac{Z \cdot p_F(\tau)}{p_B(\tau|x)R(x)}\right)^2\right], \tag{3}$$

in which p_E is an *exploratory* policy fully supported on the trajectories in G. Other loss functions, e.g., sub-trajectory balance Madan et al. (2022) and detailed balance Bengio et al. (2023), have also been studied. We refer the reader to Viviano et al. (2025) for a modular implementation of these objectives and standard benchmarks. By letting θ be the parameters of our models for p_F, p_B and Z_θ, we define

$$\mathcal{L}_{\text{TB}}(\theta; \tau) = \left(\log \frac{Z_\theta \cdot p_F(\tau; \theta)}{p_B(\tau|x; \theta)R(x)}\right)^2 \tag{4}$$

If context is clear, we will often exclude θ from the notations of p_F and p_B to avoid notational clutter. Also, notice that a GFlowNet induces a reward function s.t., for each $x \in \mathcal{X}$,

$$\hat{R}_\theta(x) = Z_\theta \cdot p_\top(x) = \mathbb{E}_{\tau\sim p_B(\cdot|x)}\left[Z_\theta \cdot \frac{p_F(\tau; \theta)}{p_B(\tau|x; \theta)}\right]. \tag{5}$$

In particular, when $\mathcal{L}_{\text{TB}}(\theta; \tau) = 0$ for all τ, the induced reward $\hat{R}_\theta(x)$ matches the true reward $R(x)$ for each $x \in \mathcal{X}$.

3 Adaptive Complementary Exploration

During training, the canonical GFlowNet may over-allocate probability mass to certain trajectories, hampering exploration of novel and high-reward regions. We illustrate this in Figure 3. When trained to sample from the RINGS distribution (see Section 4) via ϵ-greedy exploration, a GFlowNet concentrates most of its probability in a single high-reward region of the state space, under-representing the second, more distant mode. To formalize this intuition, we define the set of over-sampled trajectories below. For clarity, we will henceforth refer to a GFlowNet as $\mathfrak{g} = (Z, p_F, p_B)$.

Definition 3.1 (Over- & Under-Allocated regions)**.** Let $\mathfrak{g} = (Z, p_F, p_B)$ be a GFlowNet and $\alpha > 0$. We define the set of *over-allocated states* with respect to α as

$$\text{OA}(\alpha, \mathfrak{g}) = \{x \in \mathcal{X} \colon \hat{R}_\mathfrak{g}(x) \geq \alpha \cdot R(x)\},$$

in which $\hat{R}_\mathfrak{g}$ is the GFlowNet's induced reward function described in Equation (5). Similarly, we define $\text{UA}(\alpha, \mathfrak{g}) = \mathcal{X} \setminus \text{OA}(\alpha, \mathfrak{g})$ as the set of states with *under-allocated* probability mass. With a slight abuse of notation, we write $\tau \in \text{OA}(\alpha, \mathfrak{g})$ (resp. $\tau \in \text{UA}(\alpha, \mathfrak{g})$) to indicate that τ starts at s_o and finishes at some $x \in \text{OA}(\alpha, \mathfrak{g})$ (resp. $x \in \text{UA}(\alpha, \mathfrak{g})$). When $\mathfrak{g}$ is clear, we will simply write $\text{OA}(\alpha)$ and $\text{UA}(\alpha)$.

Our objective is to learn an exploratory policy sampling high-reward states in $\text{UA}(\alpha)$ while avoiding trajectories in $\text{OA}(\alpha)$. This can be achieved by enforcing the *DTB* condition (Definition 3.2). Given the terminology, we denote the exploration GFlowNet as $\mathfrak{g}_\nabla = (Z_\nabla, p_F^\nabla, p_B^\nabla)$.

Definition 3.2 (DTB)**.** Let $\mathfrak{g}$ and $\mathfrak{g}_\nabla$ be GFlowNets. We define the *divergent trajectory balance* (DTB) of $\mathfrak{g}_\nabla$ *with respect to* $\mathfrak{g}$ for a threshold $\alpha > 0$ and exponent $\beta > 0$ as

$$Z_\nabla \cdot p_F^\nabla(\tau) = R(x)^\beta \cdot p_B^\nabla(\tau|x); \quad \text{if } \tau \in \text{UA}(\alpha, \mathfrak{g}),$$
$$Z_\nabla \cdot p_F^\nabla(\tau) = 0; \quad \text{otherwise.}$$

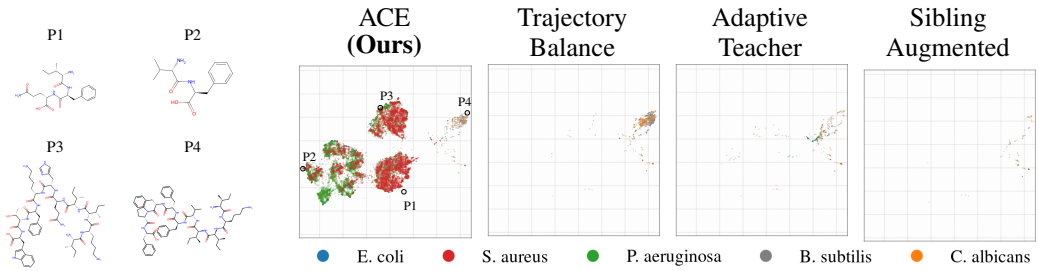


Figure 2: **Peptide embeddings colored by predicted micro-organism group.** Each dot represents the 2D projection of the k-mer embedding of the antimicrobial peptide (AMPs) in Figure 7. As we can see, ACE generates a substantially more diverse set of AMPs with likely antimicrobial activity than alternative methods. We showcase a subset of ACE's generated peptides in in the leftmost panel.

As in Definition 3.1, we will often omit α, β, and $\mathfrak{g}$ when referring to the DTB of $\mathfrak{g}_\nabla$. Similarly to prior work Madan et al. (2025); Kim et al. (2025b), we train the exploration GFlowNet on a tempered reward function to facilitate state space navigation, a technique that has been empirically shown to be effective Zhou et al. (2023) and is rooted in the literature of simulated annealing for Markov chain Monte Carlo Kirkpatrick et al. (1983). Intuitively, the DTB prunes the support of p_F^∇ to the set of trajectories with under-allocated probability mass by the canonical GFlowNet ($\mathfrak{g}$).

In order to learn an exploration GFlowNet $\mathfrak{g}_\nabla$ abiding by the conditions in Definition 3.2, we design a loss function we can optimize via gradient descent. Towards this goal, we notice that the DTB conditions can be rewritten as

$$\frac{Z_\phi^\nabla \, p_F^\nabla(\tau;\phi)}{R(x)\, p_B^\nabla(\tau;\phi)} \; + \; \mathbb{I}[\tau \in \mathrm{OA}(\alpha, \mathfrak{g})] = 1, \quad \forall \tau. \tag{6}$$

Drawing on this, we define the *divergent trajectory balance loss* $\mathcal{L}_\nabla$ as the log-squared residual between the left- and right-hand sides of Equation (6)—analogously to the TB Malkin et al. (2022) and SubTB Madan et al. (2022) losses.

Definition 3.3 (DTB Loss)**.** Let $\mathfrak{g}$ and $\mathfrak{g}_\nabla$ be GFlowNets. We define the *DTB loss* $\mathcal{L}_\nabla(\mathfrak{g}_\nabla; \tau, \alpha)$ of the exploration GFlowNet $\mathfrak{g}_\nabla$ for a trajectory τ and threshold $\alpha > 0$ as

$$\left(\log \left(\frac{Z_\phi^\nabla \, p_F^\nabla(\tau;\phi)}{R(x)^\beta \, p_B^\nabla(\tau;\phi)} + \mathbb{I}[\tau \in \mathrm{OA}(\alpha)] \right) \right)^2. \tag{7}$$

We also define $\mathcal{L}_\nabla(\mathfrak{g}_\nabla; \mathfrak{g}, \alpha) = \mathbb{E}_{\tau \sim p_F^{\epsilon,\nabla}}[\mathcal{L}_\nabla(\mathfrak{g}_\nabla; \tau, \alpha)]$ as the average of $\mathcal{L}_\nabla$ with respect to the ϵ-greedy version $p_F^{\epsilon,\nabla}$ of p_F^∇, wherein we make the dependence of $\mathcal{L}_\nabla$ on the canonical GFlowNet $\mathfrak{g}$ (via the set OA) explicit.

When optimized to zero, $\mathcal{L}_\nabla$ drives the exploratory policy p_F^∇ to sample proportionally to the reward in undersampled areas (i.e., $\mathrm{UA}(\alpha)$). We formalize this in Proposition 3.4.

Proposition 3.4 (Complementary Sampling Property)**.** *Assume $\mathcal{L}_\nabla(\mathfrak{g}_\nabla; \tau, \alpha) = 0$ for each trajectory τ starting at s_o and finishing at $\mathcal{X}$ and $\mathrm{UA}(\alpha) \neq \emptyset$. Then, the marginal $p_\top^\nabla$ of p_F^∇ over $\mathcal{X}$ is*

$$p_\top^\nabla(x) \propto R(x)^\beta \cdot \mathbb{I}[x \in \mathrm{UA}(\alpha)],$$

with normalizing constant $Z_\nabla = \sum_{x \in \mathrm{UA}(\alpha)} R(x)^\beta$.

From an information-theoretic perspective, minimizing the expectation of $\mathcal{L}_\nabla$ under a measure μ supported on trajectories in $\mathrm{OA}(\alpha)$ may be interpreted as maximizing a Kullback-Leibler divergence Kullback & Leibler (1951) based on μ.

Proposition 3.5 (Repulsive Bound)**.** *Let μ be a probability measure over trajectories supported on $\mathrm{OA}(\alpha)$, and define $p_B^\nabla(\tau) = \pi(x) p_B^\nabla(\tau|x)$ as the backward trajectory probability, with $\pi(x) \propto R(x)$ as the normalized target. Then,*

$$\mathbb{E}_{\tau \sim \mu} \left[\left(\log \frac{p_F^\nabla(\tau)}{p_B^\nabla(\tau)} + \mathbb{I}[\tau \in \mathrm{OA}(\alpha)] \right)^2 \right] \geq \left(\log(2) + \mathcal{D}_{KL}[\mu \| p_B^\nabla] - \mathcal{D}_{KL}[\mu \| p_M^\nabla] \right)^2, \tag{8}$$

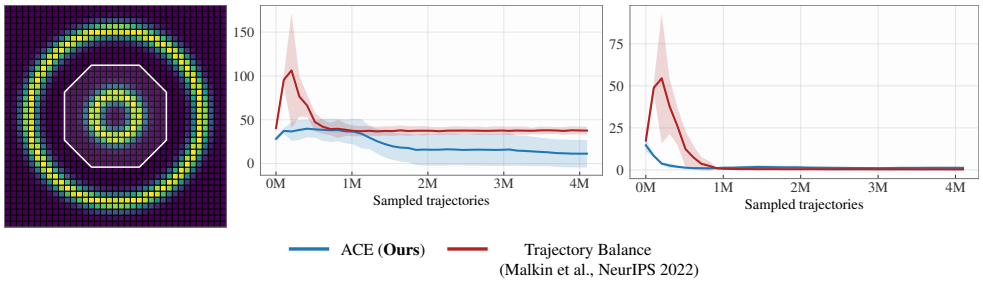


Figure 3: A GFlowNet trained on the RINGS distribution (left) via ϵ-greedy exploration may overdraw samples from a well-approximated region (polygon), misrepresenting other high-probability regions. The TB residual on the rightmost panel for the inner (center) and outer (right) rings shows ACE avoids this issue.

in which $p_M^\nabla = \frac{1}{2}(p_F^\nabla + p_B^\nabla)$ *is an uniform mixture of* p_F^∇ *and* p_B^∇*. This quantity is minimized when the marginal distribution* $p_\top^\nabla$ *of* p_F^∇ *on* $\mathcal{X}$ *vanishes on* $\mathrm{OA}(\alpha)$*.*

Adaptive Complementary Exploration. As mentioned earlier, we train the canonical GFlowNet $\mathfrak{g}$ on samples from both $\mathfrak{g}$ and $\mathfrak{g}_\nabla$ by generating trajectories from the mixture

$$p_F^{\mathrm{ACE}}(s_o, \cdot) = w \cdot p_F(s_o, \cdot) + (1 - w) \cdot p_F^{\epsilon,\nabla}(s_o, \cdot). \tag{9}$$

This raises the question: how to choose w to better emphasize diverse high-reward states during training? Intuitively, we would like $w \to 1$ as the canonical GFlowNet $\mathfrak{g}$ covers a progressively large portion of the high-probability regions in the state space, and that $w < 0.5$ when $\mathfrak{g}_\nabla$ concentrates most of the probability mass in $\mathcal{X}$. By construction, under the light of Bengio et al. (2023, Proposition 10) and Proposition 3.4, we notice that the learned Z and Z_∇ serve as proxies for the reward masses under $\mathfrak{g}$ and $\mathfrak{g}_\nabla$, respectively. With this in mind, we define $w = \frac{Z}{Z_\nabla + Z}$. as the *relative mass* under $\mathfrak{g}$,

As we will show in Proposition 3.9, such a choice satisfies the desiderata above. Based on this, we define the loss function for the canonical GFlowNet below.

Definition 3.6 (Canonical Loss)**.** Let $\mathfrak{g}$ and $\mathfrak{g}_\nabla$ be the canonical and exploration GFlowNets. We define the *canonical loss* as the trajectory balance loss averaged over the mixture distribution p_F^{ACE} in Equation (9), i.e.,

$$\mathcal{L}_{\mathrm{CAN}}(\mathfrak{g}; \mathfrak{g}_\nabla) = \mathrm{sg}(w) \cdot \mathbb{E}_{\tau \sim p_F}\left[\mathcal{L}_{\mathrm{TB}}(\mathfrak{g}; \tau, R)\right] + \mathrm{sg}(1 - w) \cdot \mathbb{E}_{\tau \sim p_F^{\epsilon,\nabla}}\left[\mathcal{L}_{\mathrm{TB}}(\mathfrak{g}; \tau, R)\right], \tag{10}$$

with sg as the stop-gradient operation—e.g., `jax.stop_gradient` in JAX Bradbury et al. (2018) or `torch.Tensor.detach` in PyTorch Paszke et al. (2019)—which detachs w from the computation graph.

We call *Adaptive Complementary Exploration* (ACE) the algorithm that learns both $\mathfrak{g}$ and $\mathfrak{g}_\nabla$ by minimizing the Canonical (Definition 3.6) and DTB (Definition 3.3) losses via Monte Carlo estimators based on their respective integrating measures. We summarize ACE in Algorithm 1. There, to determine whether $\tau \in \mathrm{OA}(\alpha)$ in Equation (7), we use a single sample from $p_B(\cdot|x)$ to estimate $\hat{R}_\mathfrak{g}(x)$.

Remark 3.7 (Notation for the loss functions). To emphasize the parameterization of our models, we also denote by $\mathcal{L}_{\mathrm{TB}}(\theta; \tau)$ and $\mathcal{L}_\nabla(\phi; \tau, \alpha)$ the losses evaluated at τ for GFlowNets $\mathfrak{g}$ and $\mathfrak{g}_\nabla$ with parameters θ and ϕ, respectively.

Notably, when $\mathfrak{g}$ and $\mathfrak{g}_\nabla$ collapse into a subset of $\mathcal{X}$, the expected on-policy gradient of $\mathcal{L}_\nabla$ pushes $\mathfrak{g}_\nabla$ towards the complement of that subset; see Proposition 3.8. Recall that we update ϕ via gradient steps in the direction of $-\nabla_\phi \mathcal{L}_\nabla$.

Proposition 3.8. *Assume* $\mathfrak{g}$ *and* $\mathfrak{g}_\nabla$ *are collapsed on a set* $\mathrm{C} \subset \mathcal{X}$*, i.e.,* $p_\top(\mathrm{C}) = 1$ *and* $p_\top^\nabla(\mathrm{C}) = 1$*, that they satisfy their respective TB conditions on all trajectories leading to* C*, and* $\alpha < 1$*. Then, if* $\mathfrak{g}_\nabla$ *is parameterized by* $\phi = (\phi_F, \phi_B, Z_\nabla)$*,* ϕ_F *and* ϕ_B *as the parameters for* p_F^∇ *and* p_B^∇*,*

$$\mathbb{E}_{\tau \sim p_F^\nabla(\cdot; \phi_F)}\left[\nabla_{\phi_F} \mathcal{L}_\nabla(\phi; \tau, \alpha)\right] = -\log(2) \nabla_{\phi_F} p_\top^\nabla(\mathrm{C}^c; \phi_F),$$

in which $p_\top^\nabla$ *is as in Equation* (1) *and* $\mathrm{C}^c := \mathcal{X} \setminus \mathrm{C}$*.*

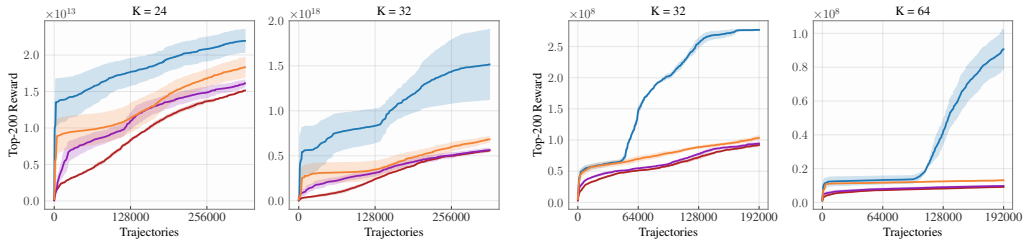


Figure 4: **ACE significantly accelerates mode-discovery for autoregressive sequence generation** with GFlowNets. Each plot shows the average reward of the unique 200 highest-valued discovered states as a function of the number of trajectories sampled throughout training.

We also demonstrate that the equilibrium of the cooperative game implemented by Algorithm 1 depends on the choice of α. If $\alpha \leq 1$, for instance, the repulsive force described in Proposition 3.5 forces $\mathfrak{g}_\nabla$ to collapse into $Z_\nabla = 0$. Ohterwise, if $\alpha > 1$, the exploration GFlowNet matches the tempered target $R(x)^\beta$ on $\mathcal{X}$.

Proposition 3.9 (Equilibrium State)**.** *Assume* $\mathfrak{g}^\star = (Z^\star, p_F^\star, p_B^\star)$ *and* $\mathfrak{g}_\nabla^\star = (Z_\nabla^\star, p_F^{\nabla,\star}, p_B^{\nabla,\star})$ *jointly satisfy*

$$\mathfrak{g}^\star = \arg\min_{\mathfrak{g}} \mathcal{L}_{\text{CAN}}(\mathfrak{g}; \mathfrak{g}_\nabla^\star) \quad \textit{and} \quad \mathfrak{g}_\nabla^\star = \arg\min_{\mathfrak{g}_\nabla} \mathcal{L}_\nabla(\mathfrak{g}_\nabla; \mathfrak{g}^\star, \alpha).$$

Then, $Z^\star := \sum_{x\in\mathcal{X}} R(x)$ *and* $p_\top^\star(x) \propto R(x)$*. When* $\alpha \leq 1$*,* $Z_\nabla^\star = 0$*. When* $\alpha > 1$*,* $Z_\nabla^\star = \sum_{x\in\mathcal{X}} R(x)^\beta$ *and* $p_\top^{\nabla,\star}(x) \propto R(x)^\beta$*, in which* $p_\top^{\nabla,\star}$ *is the marginal distribution over* $\mathcal{X}$ *induced by* $\mathfrak{g}_\nabla^\star$ *(recall Equation 1).*

Together, these results establish ACE as a principled approach for enhanced exploration of reward-dense regions during GFlowNet training. Importantly, the next section shows that ACE also consistently outperforms prior art on standard metrics used in the GFlowNet literature.

4 Experiments

In this section, we present a comprehensive empirical analysis of our method, specifically designed to address two central research questions. First (**RQ1**), we investigate whether ACE significantly increases the rate at which diverse and high-reward states are discovered during learning. Second (**RQ2**), we evaluate if ACE accelerates learning convergence. We answer both in the affirmative by measuring the top-K average reward of unique states found throughout training Pan et al. (2023); Madan et al. (2022; 2025), the convergence rate of the log-partition function of the canonical GFlowNet, and the total variation (TV) distance between the learned and target distributions, defined as $\text{TV}(p_\top, \pi) := \frac{1}{2}\sum_{x\in\mathcal{X}} |p_\top(x) - \pi(x)|$. in which $\pi(x) \propto R(x)$ is the normalized target and $p_\top$ is the GFlowNet's marginal over terminal states; see Equation (1).

Collectively, our experiments confirm that ACE is an effective algorithm that drastically improves the sample efficiency of GFlowNets. We refer the reader to Appendix C for further details on our experimental setup, where we provide a comprehensive comparison and in-depth discussion of our baselines. These include the standard Trajectory Balance (TB) Malkin et al. (2022) and recent novelty-promoting strategies such as Adaptive Teachers (AT) Kim et al. (2025b), which uses a weighted average between the reward and the TB residual, and Sibling Augmented (SA) GFlowNets Madan et al. (2025), which incorporates intrinsic rewards via Random Network Distillation (RND).

Lazy Random Walk. The state space $\mathcal{S}$ is $[[-m, m]]^d \times \{1, \ldots, T-1\}$ for $m, d, T \in \mathbb{N}$ with $md \leq T$, and $\mathcal{X} := [[-m, m]]^d \times \{T\}$, and $[[-m, m]] = \{-m, -m+1, \ldots, m\}$. The initial state is $s_o = (\mathbf{0}_d, 1) = ([0, \ldots, 0], 1)$ and each transition at $(\mathbf{s}, t)$ corresponds to either adding 1 or -1 to a chosen coordinate of $\mathbf{s}$ or staying in place; in either case, the counter t is incremented to $t+1$.

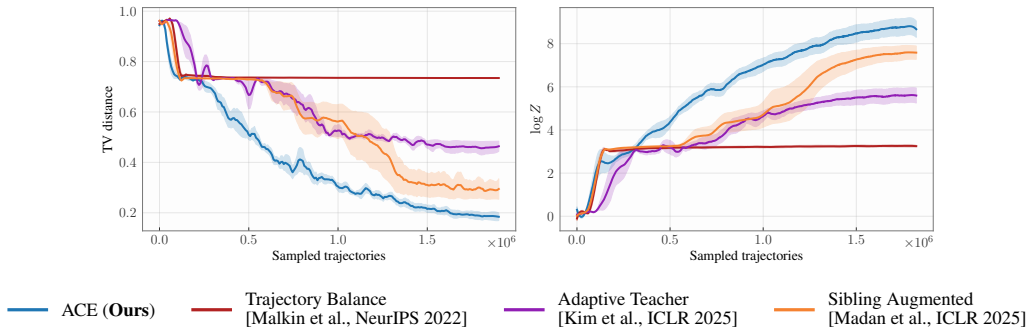


Figure 5: **ACE achieves the best goodness-of-fit** to the target distribution in the grid world task.

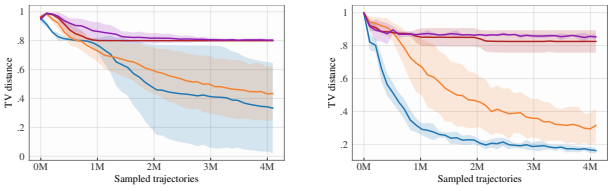
This is repeated until $t = T$ (full details in Section C). We let $\alpha = 0.2$. In particular, we assess ACE on both RINGS—shown in Figure 3—and 8 GAUSSIANS distributions; Figure 6 highlights ACE achieves the best distributional fit.

Grid world. Malkin et al. (2022); Madan et al. (2025) To further gauge ACE, we consider the standard grid world environment. There, $\mathcal{S} = [[0, H]]^d$ and $\mathcal{X} = \mathcal{S} \times \{\top\}$, in which $\top$ is an indicator of finality. The sampler starts at $s_o = \mathbf{0}$, and at each state s we either add 1 to a coordinate of s or transition to $x := s \times \{\top\} \in \mathcal{X}$. We let $H = 16$, $d = 2$, $y(x) = |^{5 \cdot x}/_H - 10|$, and train a GFlowNet to sample from $R(x) = 10^{-3} + 3 \cdot \prod_{1 \le i \le d} [y(x_i) \in (6, 8)]$.

in which $[C]$ represents Iverson's bracket, which evaluates to 1 if the clause C is true and 0 otherwise; see Figure 1. Differently from Lazy Random Walk, the stopping action poses additional exploration challenges, as a randomly initialized sampler is less likely to encounter the distant (in Euclidean norm) modes from s_o Shen et al. (2023). Notably, Figure 5 shows that ACE results in faster training convergence than both AT, SA, and ϵ-greedy GFlowNets.

Bit sequences. Malkin et al. (2022); Madan et al. (2022) We define $\mathcal{S} = \bigcup_{k \le K-1} \{1, 0\}^k$ and $\mathcal{X} = \{1, 0\}^K$ for a given sequence size K. As in Malkin et al. (2022), we let $\mathcal{M} \subseteq \mathcal{X}$ be a set of modes and $\log R(x) = \frac{1}{T} (1 - \min_{m \in \mathcal{M}} {}^{d(x,m)}/_K)$, in which $d(x, m)$ represents Levenshtein's distance between binary strings x and m and $T = {}^1/_{20}$. Concretely, $\mathcal{S}$ represents the space of bit sequences with size up to $K - 1$. Starting at $s_o = []$, we append either 1 or 0 to the current state until it reaches the size of K. We consider $K \in \{32, 64\}$. Notably, Figure 4b shows that our method finds diverse high-reward states significantly faster than baselines.

Sequence design. Silva et al. (2025) Similarly, $\mathcal{S} = \bigcup_{k \le K-1} \mathcal{V}^k$ for a finite vocabulary $\mathcal{V}$ and K, and $\mathcal{X} = \mathcal{V}^K$ with size $V := |\mathcal{V}|$. We consider $(K, V) \in \{(24, 6), (32, 4)\}$, and define the reward function of a $x \in \mathcal{X}$ through $\log R(x) := \sum_{k=1}^{K} u(k) \cdot v(x_k)$, in which $u \colon [K] \to \mathbb{R}$ and $v \colon \mathcal{V} \to \mathbb{R}$ are utility functions picked at random prior to training. (Recall $[K] = \{1, \ldots, K\}$). Remarkably, Figure 4a confirms that ACE significantly increase the discovery rate of high-reward regions for this task.



RINGS. 8 GAUSSIANS.

Figure 6: **ACE results in faster learning convergence** than prior art for GFlowNet exploration for the Lazy Random Walk task. Please consult Figure 5 below for the legend.

Bag generation. Shen et al. (2023); Jang et al. (2024) A bag is a multiset with elements taken from a set $\mathcal{V}$. In Figure 8a, we let $\mathcal{S} := \{B \subseteq_{\text{multi}} \mathcal{V} \colon |\mathcal{B}| < S\}$ and $\mathcal{X} := \{B \subseteq_{\text{multi}} \mathcal{V} \colon |\mathcal{B}| = S\}$ be the set of S-sized multi-subsets of a set $\mathcal{V}$ with size K, and define $R(x) = \sum_{e \in x} u(e)$ for an utility function $u \colon \mathcal{V} \to \mathbb{R}_+$ drawn from a geometric Gaussian process. Once again, ACE improves upon prior approaches in terms of the speed with which high-probability regions are discovered.

Quadratic Knapsack. We present results for the Quadratic Knapsack task, which is an NP-hard problem Caprara et al. (1999). Briefly, we let W be the maximum weight, $\mathbf{w} \in \mathbb{R}_+^K$ be the weight of each

of the K items, $\mathbf{u} \in \mathbb{R}_+^K$ be their utilities, and $\mathbf{A} \in \mathbb{R}^{K \times K}$ be a symmetric matrix measuring the substitutability or complementarity of each item pair. We may choose up to L copies of each item, and our objective is to find the item multiplicities $\mathbf{m} = (m_1, \ldots, m_K)$ maximizing the collection's utility, i.e.,

$$\max_{\mathbf{m} \in [[0, L]]^K} \langle \mathbf{u}, \mathbf{m} \rangle + \mathbf{m}^\top A \mathbf{m} \text{ s.t. } \langle \mathbf{m}, \mathbf{w} \rangle \leq W. \tag{11}$$

We let $K \in \{128, 256\}$ and $W = 60$. Our generative process starts at $s_o = \mathbf{0} \in \mathbb{R}^K$, and at each step we add an item to the current state s until no items can be added (either due to repetition or weight limit). In this setting, $\mathcal{S} = \{\mathbf{m} \in [[0, L]]^K : \exists k, \langle \mathbf{m}, \mathbf{w} \rangle + \mathbf{w}_k \leq W \text{ and } \mathbf{m}_k + 1 \leq L\}$, and $\mathcal{X}$ is defined as the set for which such a k does not exist, i.e., for which no more items can be added to the collection. The reward function R being defined as the objective function in Equation (11). As previously noted, ACE exhibits the best sample efficiency in the search for high-valued feasible solutions for the Quadratic Knapsack problem.

Antimicrobial Peptides (AMPs). Trabucco et al. (2022); Jain et al. (2022) We also evaluate ACE on the task of *de novo* design of AMPs potentially active against the following pathogens: *E. coli*, *S. aureus*, *P. aeruginosa*, *B. subtilis*, and *C. albicans*. The peptide design space is restricted to sequences with up to 10 amino acids. Given the standard proteinogenomic alphabet, consisting of 20 standard amino acids, this results in a search space with 10^{13} candidates.

Formally, the initial state is $s_o = []$, the state space $\mathcal{S}$ consists of all amino acid sequences of size up to $L = 10$, and $\mathcal{X} = \{s \oplus \langle \text{EOS} \rangle : s \in \mathcal{S}\}$, in which $\langle \text{EOS} \rangle$ is a special end-of-sequence token and $\oplus$ represents the concatenation operator; for ACE, we let $\alpha = 0.2$ and $\beta = 1$. The reward is derived from a Random Forest classifier Pedregosa et al. (2011); Dall'Antonia et al. (2025) predicting antimicrobial activity across pathogens (full details in Section C); we use a cutoff $c = 0.95$ and call a sequence a *mode* if its predicted activity probability satisfies $p(s) \geq c$.

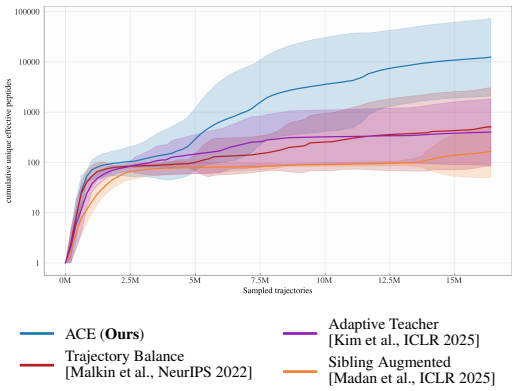


Figure 7: **ACE identifies a larger number of high-fitness AMPs** than prior methods. The plot shows the number of unique AMPs found during training with at least 95% probability of exhibiting antimicrobial activity as a function of the number of trajectories.

Crucially, Figure 7 shows that ACE discovers substantially more unique high-reward AMPs throughout training than all competing methods. On the log-scaled y-axis, ACE achieves an order-of-magnitude improvement in the cumulative number of unique effective peptides, indicating both faster mode discovery and sustained exploration. To further assess diversity, Figure 2 visualizes the modes found during training via a 2D UMAP projection of their k-mer frequency embeddings, highlighting the significantly broader coverage achieved by ACE.

5 Discussion

We introduced *Adaptive Complementary Exploration* (ACE) as a principled algorithm for effective exploration of underexplored regions during the training of GFlowNets. While former approaches focused on learning an exploratory policy via curiosity-driven methods, which we have shown may overemphasize well-approximated regions of the state space (e.g., Figure 1), ACE promotes the visitation of novel states through the newly proposed *divergent trajectory balance* (DTB) loss. Importantly, we proved that the minimization of DTB pushes the exploratory policy away from oversampled trajectories by the canonical GFlowNet, providing a rigorous foundation for our method.

Our experiments demonstrated ACE consistently and significantly outperformed prior approaches for improved GFlowNet exploration in terms of the discovery rate of diverse, high-reward regions and the goodness-of-fit to the target distribution. In conclusion, we also believe that exploring whether a non-stationary reward (e.g., curiosity-driven) can be used in Definition 3.2 and how to optimally weight the GFlowNets' samples in Equation (9) are promising directions for future research.

REFERENCES

Lazar Atanackovic and Emmanuel Bengio. Investigating generalization behaviours of generative flow networks, 2024. URL https://arxiv.org/abs/2402.05309.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In *NeurIPS (NeurIPS)*, 2021.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J. Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research (JMLR)*, 2023.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation, 2018. URL https://arxiv.org/abs/1810.12894.

Alberto Caprara, David Pisinger, and Paolo Toth. Exact solution of the quadratic knapsack problem. *INFORMS Journal on Computing*, 11(2):125–137, 1999.

Pedro Dall'Antonia, Tiago da Silva, Daniel Augusto de Souza, César Lincoln C. Mattos, and Diego Mesquita. Boosted gflownets: Improving exploration via sequential learning, 2025. URL https://arxiv.org/abs/2511.09677.

Tristan Deleu, António Góis, Chris Chinenye Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian structure learning with generative flow networks. In *UAI*, 2022.

Tristan Deleu, Mizu Nishikawa-Toomey, Jithendaraa Subramanian, Nikolay Malkin, Laurent Charlin, and Yoshua Bengio. Joint Bayesian inference of graphical structure and parameters with a single generative flow network. In *Advances in Neural Processing Systems (NeurIPS)*, 2023.

Edward J. Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, and et al. Amortizing intractable inference in large language models, 2023.

Rui Hu, Yifan Zhang, Zhuoran Li, and Longbo Huang. Beyond squared error: Exploring loss design for enhanced training of generative flow networks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4NTrco82W0.

Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure F. P. Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, Lena Simine, Payel Das, and Yoshua Bengio. Biological sequence design with GFlowNets. In *International Conference on Machine Learning (ICML)*, 2022.

Moksh Jain, Tristan Deleu, Jason Hartford, Cheng-Hao Liu, Alex Hernandez-Garcia, and Yoshua Bengio. Gflownets for ai-driven scientific discovery. *Digital Discovery*, 2(3):557–577, 2023. ISSN 2635-098X. doi: 10.1039/d3dd00002h. URL http://dx.doi.org/10.1039/D3DD00002H.

Hyosoon Jang, Minsu Kim, and Sungsoo Ahn. Learning energy decompositions for partial inference in GFlownets. In *The Twelfth International Conference on Learning Representations*, 2024.

Hohyun Kim, Seunggeun Lee, and Min hwan Oh. Symmetry-aware gflownets, 2025a. URL https://arxiv.org/abs/2506.02685.

Minsu Kim, Sanghyeok Choi, Taeyoung Yun, Emmanuel Bengio, Leo Feng, Jarrid Rector-Brooks, Sungsoo Ahn, Jinkyoo Park, Nikolay Malkin, and Yoshua Bengio. Adaptive teachers for amortized samplers. *International Conference on Learning Representations (ICLR)*, 2025b.

Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598), 1983.

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

Salem Lahlou, Tristan Deleu, Pablo Lemos, Dinghuai Zhang, Alexandra Volokhova, Alex Hernández-García, Léna Néhale Ezzine, Yoshua Bengio, and Nikolay Malkin. A theory of continuous generative flow networks. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pp. 18269–18300. PMLR, 2023.

Elaine Lau, Nikhil Vemgal, Doina Precup, and Emmanuel Bengio. Dgfn: Double generative flow networks, 2023. URL https://arxiv.org/abs/2310.19685.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. Learning gflownets from partial episodes for improved convergence and stability. In *International Conference on Machine Learning*, 2022.

Kanika Madan, Alex Lamb, Emmanuel Bengio, Glen Berseth, and Yoshua Bengio. Towards improving exploration through sibling augmented gflownets. In *International Conference on Representation Learning (ICLR)*, pp. 89636–89654, 2025.

Idriss Malek, Aya Laajil, Abhijith Sharma, Eric Moulines, and Salem Lahlou. Loss-guided auxiliary agents for overcoming mode collapse in gflownets, 2026. URL https://arxiv.org/abs/2505.15251.

Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in GFlownets. In *NeurIPS (NeurIPS)*, 2022.

Nikolay Malkin, Salem Lahlou, Tristan Deleu, Xu Ji, Edward Hu, Katie Everett, Dinghuai Zhang, and Yoshua Bengio. GFlowNets and variational inference. *International Conference on Learning Representations (ICLR)*, 2023.

Ling Pan, Nikolay Malkin, Dinghuai Zhang, and Yoshua Bengio. Better training of GFlowNets with local credit and incomplete trajectories. In *International Conference on Machine Learning (ICML)*, 2023.

Ling Pan, Moksh Jain, Kanika Madan, and Yoshua Bengio. Pre-training and fine-tuning generative flow networks. In *The Twelfth International Conference on Learning Representations*, 2024.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Max W. Shen, Emmanuel Bengio, Ehsan Hajiramezanali, Andreas Loukas, Kyunghyun Cho, and Tommaso Biancalani. Towards understanding and improving gflownet training. In *International Conference on Machine Learning*, 2023.

Tiago Silva, Rodrigo Barreto Alves, Eliezer de Souza da Silva, Amauri H Souza, Vikas Garg, Samuel Kaski, and Diego Mesquita. When do GFlownets learn the right distribution? In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=9GsgCUJtic.

Daniil Tiapkin, Nikita Morozov, Alexey Naumov, and Dmitry Vetrov. Generative flow networks as entropy-regularized rl, 2024.

Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-bench: Benchmarks for data-driven offline model-based optimization, 2022. URL https://arxiv.org/abs/2202.08450.

Nikhil Vemgal, Elaine Lau, and Doina Precup. An empirical study of the effectiveness of using a replay buffer on mode discovery in gflownets, 2023. URL https://arxiv.org/abs/2307.07674.

Siddarth Venkatraman, Moksh Jain, Luca Scimeca, Minsu Kim, Marcin Sendera, Mohsin Hasan, Luke Rowe, Sarthak Mittal, Pablo Lemos, Emmanuel Bengio, Alexandre Adam, Jarrid Rector-Brooks, Yoshua Bengio, Glen Berseth, and Nikolay Malkin. Amortizing intractable inference in diffusion models for vision, language, and control, 2024. URL https://arxiv.org/abs/2405.20971.

Joseph D. Viviano, Omar G. Younis, Sanghyeok Choi, Victor Schmidt, Yoshua Bengio, and Salem Lahlou. torchgfn: A pytorch gflownet library, 2025. URL https://arxiv.org/abs/2305.14594.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023.

Tianshu Yu. Secrets of gflownets' learning behavior: A theoretical study, 2025. URL https://arxiv.org/abs/2505.02035.

David W Zhang, Corrado Rainone, Markus Peschl, and Roberto Bondesan. Robust scheduling with gflownets. In *International Conference on Learning Representations (ICLR)*, 2023a.

Dinghuai Zhang, Hanjun Dai, Nikolay Malkin, Aaron Courville, Yoshua Bengio, and Ling Pan. Let the flows tell: Solving graph combinatorial optimization problems with gflownets. In *NeurIPS (NeurIPS)*, 2023b.

Ni Zhang and Zhiguang Cao. Hybrid-balance gflownet for solving vehicle routing problems, 2025. URL https://arxiv.org/abs/2510.04792.

Mingyang Zhou, Zichao Yan, Elliot Layne, Nikolay Malkin, Dinghuai Zhang, Moksh Jain, Mathieu Blanchette, and Yoshua Bengio. Phylogfn: Phylogenetic inference with generative flow networks, 2023.

Heiko Zimmermann, Fredrik Lindsten, Jan-Willem van de Meent, and Christian A. Naesseth. A variational perspective on generative flow networks. *Trans. Mach. Learn. Res.*, 2023, 2023.

A PROOFS

We provide rigorous proofs for each of our statements in this section.

A.1 PROOF OF PROPOSITION 3.4

We will demonstrate that $\mathcal{L}_\nabla(\mathfrak{g}_\nabla; \tau, \alpha) = 0$ for every τ if and only if p_F^∇ is supported on trajectories τ resulting in the under-allocated set $\mathrm{UA}(\alpha, \mathfrak{g})$, and that $p_\top^\nabla(x) \propto R(x)^\beta$ for x in such a set.

To see this, we rewrite $\mathcal{L}_\nabla$ as

$$\lambda \mathbb{E}_{\tau \sim p_F^{\epsilon,\nabla}} \left[\left(\log \frac{Z_\nabla p_F^\nabla(\tau)}{R(x) p_B^\nabla(\tau|x)} \right)^2 \middle| \tau \in \mathrm{UA}(\alpha) \right] + (1-\lambda) \mathbb{E}_{\tau \sim p_F^{\epsilon,\nabla}} \left[\left(\log \left(\frac{Z_\nabla p_F^\nabla(\tau)}{R(x) p_B^\nabla(\tau|x)} + 1 \right) \right)^2 \middle| \tau \in \mathrm{OA}(\alpha) \right],$$

in which $\lambda \coloneqq p_\top^{\epsilon,\nabla}(\mathrm{UA}(\alpha)) \geq \epsilon p_\top^U(\mathrm{UA}(\alpha)) > 0$, with $p_\top^U$ denoting the marginal distribution over $\mathcal{X}$ induced by the uniform policy. The first member of the above equation is globally minimized when $Z_\nabla p_F^\nabla(\tau) = R(x) p_B^\nabla(\tau|x)$ for $\tau \in \mathrm{UA}(\alpha)$, which in particular yields $Z_\nabla \neq 0$ as $\mathrm{UA}(\alpha)$ is non-empty by assumption. When $Z_\nabla \neq 0$, the second member is minimized when $p_F^\nabla(\tau) = 0$ if $\tau \in \mathrm{UA}(\alpha)$. As $x \mapsto x^2$ is non-negative, this is the only global minimizer of the DTB loss.

Under these circumstances, the marginal distribution of $\mathfrak{g}_\nabla$ over $\mathcal{X}$ is

$$\mathbb{I}[x \in \mathrm{UA}(\alpha)] \sum_{\tau \colon s_o \rightsquigarrow x} p_F^\nabla(\tau) = \mathbb{I}[x \in \mathrm{UA}(\alpha)] \cdot \sum_{\tau \colon s_o \rightsquigarrow x} \frac{R(x)^\beta}{Z} p_B(\tau|x) \propto \mathbb{I}[x \in \mathrm{UA}(\alpha)] \cdot R(x)^\beta,$$

(by Malkin et al. (2022, Proposition 1)) with normalizing constant $Z_\nabla = \sum_{x \in \mathcal{X}} R(x)^\beta$. Importantly, the above computation is valid since, by Definition 3.1, $x \in \mathrm{OA}(\alpha)$ implies that $\tau \in \mathrm{OA}(\alpha)$ for each τ going from s_o to x.

A.2 PROOF OF PROPOSITION 3.5

We will first demonstrate that, for any measure μ supported on $\mathrm{OA}(\alpha)$,

$$\mathbb{E}_{\tau \sim \mu} \left[\left(\log \left(\frac{p_F(\tau)}{\pi(x) p_B(\tau|x)} + \mathbb{I}[\tau \in \mathrm{OA}(\alpha)] \right) \right)^2 \right] \geq \left(\log(2) + \mathcal{D}_{\mathrm{KL}}\left[\mu || p_B\right] - \mathcal{D}_{\mathrm{KL}}[\mu || p_M] \right)^2, \tag{12}$$

in which $p_B(\tau) \coloneqq \pi(x) p_B(\tau|x)$ is the probability distribution over trajectories induced by p_B and the target $\pi(x) \propto R(x)$ and $p_M(\tau) \coloneqq {}^1\!/_2 p_F(\tau) + {}^1\!/_2 p_B(\tau)$ is the arithmetic average between p_F and p_B. Also, $\mathcal{D}_{\mathrm{KL}}$ is the standard Kullback-Leibler divergence, defined as

$$\mathcal{D}_{\mathrm{KL}}[p||q] = \mathbb{E}_{\tau \sim p} \left[\log \frac{p(\tau)}{q(\tau)} \right].$$

To understand Equation (12), notice that $\mathbb{I}[\tau \in \mathrm{OA}(\alpha)] = 1$ μ-almost surely and

$$\begin{aligned}
\mathbb{E}_{\tau \sim \mu} \left[\log \left(\frac{p_F(\tau)}{p_B(\tau)} + 1 \right) \right] &= \mathbb{E}_{\tau \sim \mu} \left[\log \left(\frac{p_F(\tau) + p_B(\tau)}{p_B(\tau)} \right) \right] \\
&= \mathbb{E}_{\tau \sim \mu} \left[\log \left(\frac{p_F(\tau) + p_B(\tau)}{p_B(\tau)} \right) + \log \frac{\mu(\tau)}{\mu(\tau)} \right] \\
&= \mathbb{E}_{\tau \sim \mu} \left[\log \left(\frac{p_F(\tau) + p_B(\tau)}{\mu(\tau)} \right) \right] + \mathbb{E}_{\tau \sim \mu} \left[\log \left(\frac{\mu(\tau)}{p_B(\tau)} \right) \right] \\
&= \log(2) - \mathbb{E}_{\tau \sim \mu} \left[\log \left(\frac{2\mu(\tau)}{p_F(\tau) + p_B(\tau)} \right) \right] + \mathbb{E}_{\tau \sim \mu} \left[\log \left(\frac{\mu(\tau)}{p_B(\tau)} \right) \right] \\
&= \log(2) - \mathcal{D}_{\mathrm{KL}}[\mu||p_M] + \mathcal{D}_{\mathrm{KL}}[\mu||p_B].
\end{aligned}$$

By Jensen's inequality and the nonnegativity of $x \mapsto \log(1+x)$ for $x \geq 0$,

$$\begin{aligned}
\mathbb{E}_{\tau \sim \mu} \left[\left(\log \left(\frac{p_F(\tau)}{p_B(\tau)} + 1 \right) \right)^2 \right] &\geq \mathbb{E}_{\tau \sim \mu} \left[\log \left(\frac{p_F(\tau)}{p_B(\tau)} + 1 \right) \right]^2 \\
&= \left(\log(2) - \mathcal{D}_{\mathrm{KL}}[\mu||p_M] + \mathcal{D}_{\mathrm{KL}}[\mu||p_B] \right)^2.
\end{aligned} \tag{13}$$

This proves our information-theoretic lower bound for the DTB loss. Clearly, when $p_F = 0$ on the support of μ,

$$\mathcal{D}_{\mathrm{KL}}[\mu||p_B] = \mathcal{D}_{\mathrm{KL}}[\mu||p_M] - \log(2),$$

which minimizes the right-hand side (RHS) of Equation (13). When $p_F(\tau) > 0$ for a certain τ for which $\mu(\tau) > 0$, $p_B(\tau) < p_F(\tau) + p_B(\tau)$ and $\log(2) + \mathcal{D}_{\mathrm{KL}}[\mu||p_B] > \mathcal{D}_{\mathrm{KL}}[\mu||p_M]$; hence, such a p_F does not minimize $(\log(2) - \mathcal{D}_{\mathrm{KL}}[\mu||p_M] + \mathcal{D}_{\mathrm{KL}}[\mu||p_B])^2$. As a consequence, $p_F = 0$ on the support $\mathrm{OA}(\alpha)$ of μ is the only minimizer of the RHS of Equation (13).

A.3 Proof of Proposition 3.8

We will demonstrate that the on-policy expected gradient of the DTB loss function pushes the exploration GFlowNet $\mathfrak{g}_\nabla$ towards the complement of the canonical GFlowNet $\mathfrak{g}$'s support when $\mathfrak{g}$'s is collapsed into a subset C of $\mathcal{X}$. For this, first notice that, since $p_\top(\mathrm{C}) = 1$, $p_F(\tau) = 0$ for each τ resulting in $\mathrm{C}^c \coloneqq \mathcal{X} \setminus \mathrm{C}$; otherwise, $p_\top(\mathrm{C}^c) \geq p_F(\tau) > 0$ and $p_\top(\mathrm{C}) = 1 - p_\top(\mathrm{C}^c) < 1$.

As in Definition 3.1, we will adopt the convention that $\tau \in \mathrm{C}$ if τ leads up to a state $x \in \mathrm{C}$. As p_F satisfies the TB condition for the trajectories in C, $Zp_F(\tau) = R(x)p_B(\tau|x)$ for $\tau \in \mathrm{C}$. By our previous observation, $p_F(\tau) = 0$ for $\tau \in \mathrm{C}^c$. As a consequence, since $\alpha < 1$, we have $\mathrm{UA}(\alpha, \mathfrak{g}) = \mathrm{C}^c$ and $\mathrm{OA}(\alpha, \mathfrak{g}) = \mathrm{C}$. As such, the loss function $\mathcal{L}_\nabla$ for $\mathfrak{g}_\nabla$ becomes

$$\mathcal{L}_\nabla(\phi; \tau, \alpha) = \begin{cases} \mathcal{L}_{\mathrm{TB}}(\phi; \tau) \text{ if } \tau \in \mathrm{C}^c, \\ \mathcal{L}_{\mathrm{SP}}(\phi; \tau) = \mathrm{softplus}\left(\log \frac{Zp_F^\nabla(\tau)}{R(x)^\beta p_B^\nabla(\tau|x)}\right)^2 \text{ otherwise,} \end{cases}$$

in which $\mathrm{softplus}$ is the mapping $x \mapsto \log(\exp\{x\} + 1)$. As p_F^∇ is collapsed into C, only the second term matters for our calculations. Also, if $Q(\phi; \tau) = \frac{Z_\nabla p_F^\nabla(\tau)}{R(x)p_B^\nabla(\tau|x)}$,

$$\nabla_{\phi_F}\mathcal{L}_{\mathrm{SP}}(\phi; \tau) = 2 \cdot \log(Q(\phi; \tau) + 1) \cdot \frac{1}{Q(\phi; \tau) + 1} \cdot \nabla_{\phi_F} Q(\phi; \tau).$$

Based on our assumptions, $Q(\phi; \tau) = 1$ for $\tau \in \mathrm{C}$; hence,

$$\begin{aligned} \nabla_{\phi_F}\mathcal{L}_{\mathrm{SP}}(\phi; \tau) &= \log(2) \cdot \nabla_{\phi_F} Q(\phi; \tau) \\ &= \log(2) \cdot \frac{Z}{R(x)p_B^\nabla(\tau|x)} \cdot \nabla_{\phi_F} p_F(\tau; \phi_F) \\ &= \log(2) \cdot \frac{1}{p_F^\nabla(\tau; \phi_F)} \cdot \nabla_{\phi_F} p_F^\nabla(\tau; \phi_F). \end{aligned}$$

In this scenario, the expectation of $\nabla_{\phi_F}\mathcal{L}_\nabla$ with respect to $p_F(\tau; \phi_F)$ is

$$\begin{aligned} \mathbb{E}_{\tau\sim p_F^\nabla(\cdot;\phi_F)}\left[\nabla_{\phi_F}\mathcal{L}_\nabla(\phi; \tau, \alpha)\right] &= \mathbb{E}_{\tau\sim p_F^\nabla(\cdot;\phi_F)}\left[\nabla_{\phi_F}\mathcal{L}_{\mathrm{SB}}(\phi; \tau)\right] \\ &= \mathbb{E}_{\tau\sim p_F^\nabla(\cdot;\phi_F)}\left[\log(2) \cdot \frac{1}{p_F^\nabla(\tau; \phi_F)} \cdot \nabla_{\phi_F} p_F^\nabla(\tau; \phi_F)\right] \\ &= \sum_{x\in\mathrm{C}} \sum_{\tau:\ s_o\rightsquigarrow x} p_F^\nabla(\tau; \phi_F) \cdot \log(2) \cdot \frac{1}{p_F^\nabla(\tau; \phi_F)} \cdot \nabla_{\phi_F} p_F^\nabla(\tau; \phi_F) \\ &= \log(2) \sum_{x\in\mathrm{C}} \sum_{\tau:\ s_o\rightsquigarrow x} \nabla_{\phi_F} p_F^\nabla(\tau; \phi_F) \\ &= \log(2) \cdot \nabla_{\phi_F} \sum_{x\in\mathrm{C}} p_\top^\nabla(x) \\ &= \log(2) \cdot \nabla_{\phi_F}\left(1 - \sum_{x\in\mathrm{C}^c} p_\top^\nabla(x; \phi_F)\right) \\ &= -\log(2)\nabla_{\phi_F} \sum_{x\in\mathrm{C}^c} p_\top^\nabla(x; \phi_F) = -\log(2) \cdot \nabla_{\phi_F} p_\top^\nabla(\mathrm{C}^c). \end{aligned}$$

This shows that, under Proposition 3.8 conditions, the on-policy expected gradient of the DTB loss for the exploration GFlowNet points in the direction of decreasing probability mass in C^c. As we optimize ϕ via gradient descent on $\mathcal{L}_\nabla$, our algorithm moves in the direction of increasing the probability mass in C^c according to the exploration GFlowNet's model.

A.4 Proof of Proposition 3.9

We fix $\mathfrak{g}_\nabla = (Z_\nabla, p_F^\nabla, p_B^\nabla)$. Then, the loss function

$$\mathcal{L}_{\mathrm{CAN}}(\mathfrak{g}; \mathfrak{g}_\nabla)$$

is minimized when $Z^\star = \sum_{x\in\mathcal{X}} R(x)$ and $Z^\star p_F^\star(\tau) = p_B^\star(\tau|x)R(x)$ for each complete trajectory $\tau\colon s_o \rightsquigarrow x$. Under these conditions, the set of trajectories with over-allocated mass,

$$\mathrm{OA}(\alpha, \mathfrak{g}^\star) = \{\tau\colon Z^\star p_F^\star(\tau) \geq \alpha R(x) p_B^\star(\tau|x)\}$$

either contains every trajectory (in case $\alpha \leq 1$) or none (if $\alpha > 1$). In the former case, the loss function for ACE reduces to

$$\mathbb{E}_{\tau\sim p_F^{\epsilon,\nabla}}\left[\left(\log\left(\frac{Z_\nabla p_F^\nabla(\tau)}{R(x)^\beta p_B(\tau|x)} + 1\right)\right)^2\right],$$

which is minimized by $Z_\nabla^\star = 0$. In the latter case, the loss function for ACE becomes

$$\mathbb{E}_{\tau\sim p_F^{\epsilon,\nabla}}\left[\left(\log\frac{Z_\nabla p_F^\nabla(\tau)}{R(x)^\beta p_B^\nabla(\tau|x)}\right)^2\right],$$

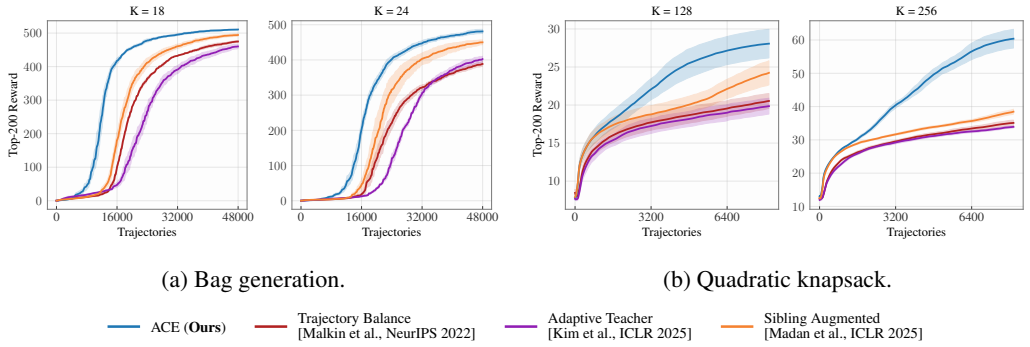
which is the standard TB loss Malkin et al. (2022) under an ϵ-greedy policy, minimized when $Z_\nabla^\star = \sum_{x\in\mathcal{X}} R(x)^\beta$ and the marginal of $p_F^{\nabla,\star}(s_o,\cdot)$ over $\mathcal{X}$ satisfies $p_\top^{\nabla,\star}(x) \propto R(x)^\beta$. Conversely, if our exploration GFlowNet satisfies either of these conditions, it should be clear by Malkin et al. (2022, Proposition 1) that the optimal canonical GFlowNet is the one satisfying $p_\top^\star(x) \propto R(x)$ and $Z^\star = \sum_{x\in\mathcal{X}} R(x)$. Indeed, we separate our demonstration into the following cases.

1. If $\alpha > 1$ and $p_\top^\nabla \propto R(x)^\beta$, the best $p_F^\star$ minimizing the GFlowNet's canonical loss in Definition 3.6 satisfies $p_\top^\star(x) \propto R(x)$ and $Z = \sum_{x\in\mathcal{X}} R(x)$ since $R(x) > 0$ is a positive measure on $\mathcal{X}$.
2. Otherwise, if $Z_\nabla^\star = 0$, then the weighting parameter $w = 1$ and the GFlowNet is trained via TB on-policy by Definition 3.6. By Proposition 3.4, $Z_\nabla = 0$ is only optimal as long as the GFlowNet $\mathfrak{g}$ maintains full support over the space of trajectories. Under this condition, the only minimizer of the canonical loss is the GFlowNet $\mathfrak{g}$ satisfying $p_\top(x) \propto R(x)$ and $Z = \sum_{x\in\mathcal{X}} R(x)$.

As such, we have shown via a fixed-point-based argument that both of these are equilibria solutions to the minimization problem stated in Proposition 3.9.

Algorithm 1 Adaptive Complementary Exploration (ACE)

Require: Reward function $R(x)$, threshold α
GFlowNets $\mathfrak{g} \leftarrow (\theta, Z_\theta)$ and $\mathfrak{g}_\nabla \leftarrow (\phi, Z_\phi^\nabla)$
while not converged **do**
  // Phase 1: Sampling
  $\mathcal{B} \leftarrow \{\tau \sim p_F(\cdot;\theta)\}$
  $\mathcal{B}_\nabla \leftarrow \{\tau \sim p_F^{\epsilon,\nabla}(\cdot;\phi)\}$
  // Phase 2: Exploitation Update
  Calculate mixing weight: $w \leftarrow \mathrm{sg}\left(\frac{Z_\theta}{Z_\theta + Z_\phi}\right)$
  $\mathcal{L}_1 \leftarrow \frac{1}{|\mathcal{B}|}\sum_{\tau\in\mathcal{B}} \mathcal{L}_{\mathrm{TB}}(\theta;\tau)$
  $\mathcal{L}_2 \leftarrow \frac{1}{|\mathcal{B}_\nabla|}\sum_{\tau\in\mathcal{B}_\nabla} \mathcal{L}_{\mathrm{TB}}(\theta;\tau)$
  $\mathcal{L} \leftarrow w\cdot\mathcal{L}_1 + (1-w)\mathcal{L}_2$
  Update θ by a gradient step on $\nabla_\theta\mathcal{L}$.
  // Phase 3: Exploration Update
  $\mathcal{L}_{\mathrm{exp}} \leftarrow \frac{1}{|\mathcal{B}_\nabla|}\sum_{\tau\in\mathcal{B}_\nabla} \mathcal{L}_\nabla(\phi;\tau,\alpha)$
  Update ϕ by a gradient step on $\nabla_\phi\mathcal{L}_{\mathrm{exp}}$.
end while



(a) Bag generation.

(b) Quadratic knapsack.

Figure 8: **ACE finds diverse and high-reward states faster** than prior approaches for improved GFlowNet exploration for the bag generation (left) and quadratic knapsack (right) problems. In both (a) and (b), K denotes the number of available items for selection.

B Related Works

Generative Flow Networks (GFlowNets; Bengio et al., 2021; 2023; Lahlou et al., 2023) have found successful applications in combinatorial optimization Zhang et al. (2023b;a), causal discovery Deleu et al. (2022; 2023), biological sequence design Jain et al. (2022), and LLM finetuning Hu et al. (2023); Venkatraman et al. (2024). They have also found promising applications in the AI for Science community Jain et al. (2023); Wang et al. (2023), as illustrated in the task for *de novo* design of AMPs in Section 4, with their relationship to variational inference and reinforcement learning being formally established by Tiapkin et al. (2024); Malkin et al. (2023); Zimmermann et al. (2023). In this context, many studies focused on improving the sample efficiency of GFlowNets (e.g., Pan et al. (2023; 2024); Hu et al. (2025); Madan et al. (2022); Zhang & Cao (2025)), while others outlined their limitations Kim et al. (2025a); Silva et al. (2025); Shen et al. (2023); Yu (2025), with the effective exploration of diverse and high-reward states often being the main empirical concern. Although previous works have previously considered training multiple GFlowNets concomitantly to speed up learning Lau et al. (2023); Madan et al. (2025); Kim et al. (2025b); Malek et al. (2026), ACE is—to the best of our knowledge—the first approach that directly promotes novelty through a penalty term that enforces a diversity-inducing balance condition, which we call Divergent Trajectory Balance.

C Experimental Details

This section provides further experimental details for our empirical analysis in Section 4.

C.1 Baselines and Exploration Strategies

In this work, we compare ACE against several established strategies for promoting exploration in GFlowNets. Beyond the standard on-policy Trajectory Balance (TB) Malkin et al. (2022), we focus on a growing body of literature that parameterizes an exploration policy p_E as a GFlowNet trained to sample from novelty-promoting modifications of the reward R.

The first baseline is the Adaptive Teachers (AT) GFlowNet. Following Kim et al. (2025b), this method trains p_E to sample from a weighted average between $R(x)$ and the loss function $\mathcal{L}_{\text{TB}}$ in log-space:

$$\log R_{\text{AT}}(x) = \log R_{\text{T}}(x) + \alpha \log R(x),$$

where $\alpha > 0$. Defining $\delta(\tau) \coloneqq \log R(x) + \log p_B(\tau|x) - \log p_F(\tau) - \log Z$ as the base residual, the novelty reward is given by:

$$\log R_{\text{T}}(x) = \mathbb{E}_{\tau \sim p_B(\cdot|x)} \left[\log \left(\epsilon + \left(1 + C \cdot \mathbf{1}_{\delta(\tau)>0} \right) \delta(\tau)^2 \right) \right]. \tag{14}$$

Another significant approach is the Sibling Augmented (SA) GFlowNet. As proposed by Madan et al. (2025), this approach introduces intrinsic rewards based on Random Network Distillation (RND)

Burda et al. (2018) as a proxy for novelty:

$$R_{\mathrm{SA}}(x;\tau) = \left(R(x)^{\beta_1} + \left(\sum_{s \in \tau} \mathrm{R_A}(s) \right)^{\beta_2} \right)^{\beta_3}, \tag{15}$$

where $\beta_1, \beta_2, \beta_3 > 0$ and $R_{\mathrm{A}}(s) = \|\psi(s) - \psi_{\mathrm{random}}(s)\|^2$ represents the RND error between a neural network ψ and a randomly fixed model ψ_{random} Pan et al. (2023); Madan et al. (2025). Note that R_{SA} is path-dependent.

While R often represents physical or statistical quantities, the novelty rewards R_{T} and R_{A} are error functions of neural networks that capture novelty only indirectly. Our method circumvents the potential difficulties of sampling from these error surfaces by directly promoting the visitation of novel states through the Divergent Trajectory Balance (DTB) loss.

C.2 Optimization & Architecture

Architecture. Across all environments we parameterize the forward and backward policies (p_F, p_B) with lightweight neural networks producing action logits.

For the *Lazy Random Walk* environment, both p_F and p_B use `FourierTimePolicy`. The policy input is $\mathrm{obs} = (x, y, \tau)$, where $(x, y) \in \mathbb{R}^2$ are the current coordinates and $\tau \in [0, 1]$ is a (clipped) normalized time variable. We construct Fourier features with frequencies $f_k = 2^k$ for $k = 0, \ldots, n_{\mathrm{freq}} - 1$:

$$\phi(\tau) = \left[\mathbf{1}_{\texttt{include\_tau}} \tau,\ \{\sin(2\pi\tau f_k)\}_k,\ \{\cos(2\pi\tau f_k)\}_k \right],$$

concatenate them with (x, y), and pass the result through an MLP with `num_layers` layers, `hidden_dim` hidden units, and ReLU activations, outputting logits over 5 discrete actions.

For *AMPs*, we use a windowed MLP policy. Given a padded sequence $s \in \{0, \ldots, V-1\}^L$ with PAD/EOS $= 0$, we compute the current length $\ell = \sum_{t=1}^{L} \mathbf{1}[s_t \neq 0]$, embed the last $W = 6$ tokens with an embedding of dimension $D = 64$, flatten the resulting $W \times D$ representation, concatenate a sinusoidal positional encoding $\mathrm{PE}(\ell) \in \mathbb{R}^{d_{\mathrm{pos}}}$ with $d_{\mathrm{pos}} = 16$, and map to logits over the vocabulary via a two-layer MLP $(WD + d_{\mathrm{pos}}) \to 128 \to V$ with ReLU.

For *all other environments*, we follow the same setup as before: p_F and p_B are parameterized by an MLP with two hidden layers and 128 hidden units per layer, using Leaky ReLU activations throughout.

Optimization Across all environments, we optimize the GFlowNet policy parameters (i.e., those of p_F and p_B) and the log-partition estimate $\log Z$ with separate optimizers, and we use AdamW (Loshchilov & Hutter, 2019) throughout.

For the *Lazy Random Walk* environment, we use learning rates 5×10^{-3} for the policy parameters and 5×10^{-2} for $\log Z$, together with a linear learning-rate schedule that decays from a factor of 1.0 to 0.1 over the training horizon applied to both optimizers.

For *AMPs*, we use fixed learning rates 0.05 for the forward policy and 0.1 for $\log Z$, without any scheduler.

For *all other environments*, we follow the same setup as before: AdamW with learning rate 10^{-2} linearly decayed to 10^{-4} for the policy parameters, and a learning rate $10\times$ larger for $\log Z$.

To ensure a fair comparison between methods, both ACE and SA GFlowNets are trained with a batch size equal to half that of AT and TB GFlowNets.

Random seeds and uncertainty bands. Unless otherwise stated, all curves report the mean over multiple random seeds and an uncertainty band corresponding to ± 1 standard deviation across seeds. For the *AMP* experiments we use 15 seeds, from 10 to 24 For the *Lazy Random Walk* experiments we use seeds 5, $\{42, 43, 44, 45, 46\}$. For *all other environments* we use 3 seeds $\{42, 126, 210\}$. For the *AMP* plots reported on a log scale, we compute the mean and standard deviation in the log-domain, so that the displayed ± 1 standard deviation band is symmetric in log space.

C.3 Environment specifications

The experimental setup for each environment was described in Section 4 in the main text. The number of training iterations was set to 5000 for sequence design, 3000 for bit sequences, 256 for knapsack, 30000 for grid world, 1500 for bags, 4000 for both lazy random walk and *AMPs*. Across all tasks, we use ϵ-greedy exploration: we set $\epsilon = 0.3$ for *AMPs*, $\epsilon = 0.1$ for *Lazy Random Walk*, and $\epsilon = 0.05$ for all other environments. We use $\beta = 1$ on *AMPs* and $\beta = 0.25$ on all remaining environments. For α, we set $\alpha = 0.2$ for *AMPs* and *Lazy Random Walk*, and $\alpha = 0.3$ for the rest.

Random forest classifier for antimicrobial activity. The proxy reward function is based on a Random Forest classifier trained per pathogen. The classifier uses 500 estimators and balanced class weights, trained on one-hot encoded sequences. Negatives were sampled uniformly to match the length distribution of the positive set, and evaluation was performed using 5-fold stratified cross-validation (ROC-AUC). Across pathogens, we obtain strong predictive performance: $\mathrm{AUC} = 0.944$ for *E. coli*, 0.942 for *S. aureus* , 0.913 for *P. aeruginosa*, 0.905 for *B. subtilis* , and 0.930 for *C. albicans*. Then, for a candidate sequence s we compute the predicted antimicrobial-activity probability for each pathogen and aggregate them as $p(s) = \max_{j \in \mathcal{P}} \Pr(y = 1 \mid s, j)$. We convert this score into a log-reward by comparing it to a cutoff $c = 0.95$ in logit space and applying temperature scaling:

$$\log R_{\mathrm{raw}}(s) = \frac{\mathrm{logit}(p(s)) - \mathrm{logit}(c)}{T}, \qquad T = 0.3.$$

If $\log R_{\mathrm{raw}}(s) < 0$, we additionally scale the penalty by the (padded) sequence length $\ell(s)$; finally, we clip $\log R(s)$ to $[-30, 0]$. In particular, sequences with $p(s) \geq c$ satisfy $\log R(s) = 0$, i.e., $R(s) = 1$.

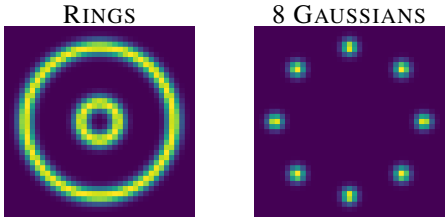


Figure 9: **Lazy Random Walk target distributions.**

Lazy Random Walk Distributions For the lazy random walk experiments we set $m = 18$ and $T = 2m$ so that its possible to traverse the full domain $[[-m, m]]^d$ within the horizon. We consider two synthetic, multimodal targets and use their (unnormalized) densities as rewards, where $x \in \mathcal{X}$ denotes the terminal position at $t = T$ and we add a small uniform floor λ to avoid zero densities. Concretely, 8 GAUSSIANS is defined as an isotropic mixture of 8 Gaussians whose means are equally spaced on a circle of radius $R = 0.8m$, $\rho_{8\mathrm{G}}(x) \propto \sum_{k=1}^{8} \exp\big(-\|x - \mu_k\|_2^2/2\big)$ with $\mu_k = (R\cos\theta_k, R\sin\theta_k)$ and $\theta_k = 2\pi(k-1)/8$; RINGS is a radial mixture over a set of radii $r_1 = 0.2m$ and $r_1 = 0.8m$ with width $\sigma_r = 1$, $\rho_{\mathrm{RINGS}}(x) \propto \sum_i \exp\big(-(\|x\|_2 - r_i)^2/(2\sigma_r^2)\big)$,

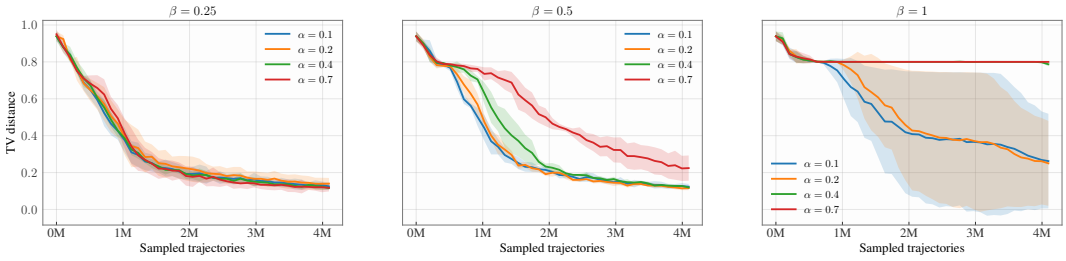


Figure 10: **Sensitivity to DTB hyperparameters.** TV distance vs. sampled trajectories on RINGS for a sweep over $\alpha \in \{0.1, 0.2, 0.4, 0.7\}$ (colors) and $\beta \in \{0.25, 0.5, 1\}$ (panels). Solid lines denote the mean across seeds and shaded regions show ± 1 standard deviation. We observe a broad stable regime for $\beta = 0.25$, whereas larger β makes training more sensitive to α and can induce failure modes for $\alpha \geq 0.4$ at $\beta = 1$ (TV plateau), together with markedly increased variance.

D HYPERPARAMETER SENSITIVITY

We evaluate the sensitivity of ACE to the DTB hyperparameters (α, β) on RINGS, where α sets the allocation threshold used to classify regions as sufficiently learned (and thus excluded from DTB enforcement), while β controls reward tempering. We sweep $\alpha \in \{0.1, 0.2, 0.4, 0.7\}$ and $\beta \in \{0.25, 0.5, 1\}$ and report the TV distance as a function of sampled trajectories. Figure 10 shows that performance is stable across a broad range of α for $\beta = 0.25$, while larger β increases sensitivity: for $\beta = 0.5$, larger α slows convergence, and for $\beta = 1$ values $\alpha \geq 0.4$ frequently lead to training failure (TV plateaus close to its initial value), with substantially higher variance even for the best-performing settings. These observations motivate our default choice of moderate β and small-to-moderate α across environments.

E IMPLEMENTATION DETAILS OF THE ACE ALGORITHM

In this section, we provide the procedural realization of the Adaptive Complementary Exploration (ACE) framework. Building upon the theoretical foundations established in the main text, the algorithm operationalizes the joint optimization of the exploitation model $\mathfrak{g}$ and the exploration model $\mathfrak{g}_\nabla$ using Monte Carlo estimators for the Canonical and DTB losses. Specifically, ACE alternates between sampling trajectories from both policies and performing gradient updates that leverage a dynamic mixing weight w, computed from the learned log-partition functions. This structure ensures that the update for θ is informed by both on-policy samples and divergent trajectories, while the update for ϕ is driven by the repulsive gradient described in Proposition 3.8, effectively pushing the exploration sampler towards the complement of the current high-reward regions.