Amortized Active Generation of Pareto Sets

Daniel M. Steinberg [1] **Asiri Wijesinghe** [1] **Rafael Oliveira** [1] **Piotr Koniusz** [1] **Cheng Soon Ong** [1] **Edwin V. Bonilla** [1]

Abstract

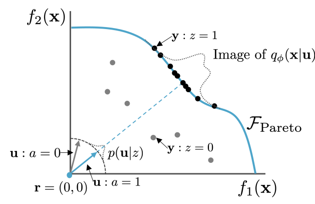
We propose active generation of Pareto sets (A-GPS), a framework for online discrete black-box multi-objective optimization that learns a generative model of the Pareto set while supporting a-posteriori preference conditioning. A-GPS avoids costly hyper-volume computations and enables flexible sampling across the Pareto front without retraining. Experiments on synthetic functions and protein design tasks show strong sample efficiency and effective preference incorporation.

1. Introduction

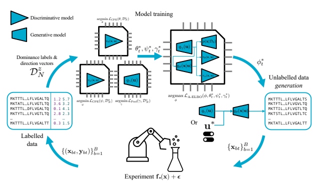
Scientific and engineering applications often require optimizing complex, high-dimensional discrete objects via expensive black-box evaluations. Examples include designing protein sequences, small molecules, or DNA constructs, where each evaluation involves costly simulations or assays, making efficient search essential. These problems commonly involve multiple, conflicting objectives, e.g., balancing thermal stability, catalytic activity, and yield in protein engineering. The set of non-dominated solutions, known as the Pareto set, captures optimal trade-offs and is critical for downstream decision-making.

Traditional multi-objective Bayesian optimization (MOBO) approaches rely on acquisition functions like expected hypervolume improvement (EHVI) (Yang et al., 2019; Daulton et al., 2020; Ament et al., 2023) or random scalarizations (Knowles, 2006; Paria et al., 2020; De Ath et al., 2022). These either scale poorly with the number of objectives or require dense sampling to capture complex Pareto fronts.

We propose a different approach inspired by multi-objective generation (MOG) methods (Yuan et al., 2025; Yao et al., 2024): we directly learn a generative model of the Pareto set in an online black-box setting. Building on variational search distributions (VSD) (Steinberg et al., 2025), which alternates between fitting a class probability estimator (CPE)



(a) Problem setup


Discriminative model
Generative model
Model training
$\operatorname{argmin}_\theta \mathcal{L}_{\text{CPE}}(\theta, \mathcal{D}_N^z)$
$\theta_t^*, \psi_t^*, \gamma_t^*$
Dominance labels & direction vectors
$\mathcal{D}_N^z$
$\operatorname{argmin}_\psi \mathcal{L}_{\text{CPE}}(\psi, \mathcal{D}_N^z)$
$\operatorname{argmin}_\gamma \mathcal{L}_{\text{Pref}}(\gamma, \mathcal{D}_N^z)$
$\operatorname{argmax}_\phi \mathcal{L}_{\text{A-ELBO}}(\phi, \theta_t^*, \psi_t^*, \gamma_t^*)$
ϕ_t^*
Unlabelled data *generation*
Or
$\mathbf{u}^*$
Labelled data
$\{(\mathbf{x}_{bt}, \mathbf{y}_{bt})\}_{b=1}^B$
$\{\mathbf{x}_{bt}\}_{b=1}^B$
Experiment $\mathbf{f}_\bullet(\mathbf{x}) + \epsilon$

(b) A-GPS flow

Figure 1: (a) Our approach to learning a generative model of Pareto sets supporting preference conditioning for which we use preference direction vectors $\mathbf{u}$ that are defined from observed or desired user's outcomes. New binary labels z and a are created, determining Pareto set membership and alignment between the design $\mathbf{x}$ and $\mathbf{u}$, respectively. (b) A-GPS learns all the the distributions involved by optimizing different components of a reverse Kullback-Leibler (KL) loss. At time t, the optimized variational distribution $q_\phi(\mathbf{x}|\mathbf{u})$ with parameters ϕ_t^* is used to generate new designs that can incorporate new user's preferences $\mathbf{u}^*$. We iterate until a convergence/user criterion is satisfied.

and updating a generative model of high-performing designs, we introduce a Pareto-set class probability estimator (CPE) to focus learning on non-dominated regions. This avoids hyper-volume computation and scalarization, yielding a scalable alternative to existing methods. To enable preference-aware sampling, we further introduce *preference direction vectors* for conditioning the model post hoc on user-specified trade-offs. Using amortized variational inference (VI), our method supports flexible, preference-guided exploration without retraining. We demonstrate that active generation of Pareto sets (A-GPS) outperforms baselines across synthetic and protein design benchmarks. Related work is discussed in Appendix D.

2. Preliminaries

In this section we briefly recall the concepts of black-box multi-objective optimization (MOO) and active generation.

2.1. Optimizing over multiple objectives

In this work we are concerned with generating discrete or mixed discrete-continuous designs, for example sequences

[1]CSIRO's Data61, Australia. Correspondence to: Daniel M. Steinberg <Daniel.Steinberg@csiro.au>. *SIM: Scaling up Intervention Models Workshop at ICML*, Vancouver, Canada, 2025. Copyright 2025 by the author(s).

$\mathbf{x} \in \mathcal{X} = \mathcal{V}^M$, where $\mathcal{V}$ is the sequence vocabulary and M is the sequence length, that have particular measurable properties $\mathbf{y} \in \mathbb{R}^L$. We assume the 'black-box' relationship $\mathbf{y} = \mathbf{f}_\bullet(\mathbf{x}) + \boldsymbol{\epsilon}$ where $\mathbf{f}_\bullet(\mathbf{x}) = [f_\bullet^1(\mathbf{x}), \ldots, f_\bullet^l(\mathbf{x}), \ldots, f_\bullet^L(\mathbf{x})]$ and $\mathbb{E}_{p(\boldsymbol{\epsilon})}[\boldsymbol{\epsilon}] = \mathbf{0}$. The black-box function $\mathbf{f}_\bullet(\cdot)$ could be an empirical observation, or an expensive physics/chemistry simulation, etc. In MOO we would like to find the set $\mathcal{S}$ such that, $\mathcal{S} = \operatorname{argmax}_{\mathbf{x}\in\mathcal{X}} \mathbf{f}_\bullet(\mathbf{x})$. Often the objectives, $\mathbf{f}_\bullet$, are in conflict with one another, and so there does not exist one unique feasible maximizer. Instead, we are interested in finding the set of designs for which we cannot increase one objective without compromising others. This is known as the Pareto set $\mathcal{S}_{\text{Pareto}} \subset \mathcal{X}$,

$$\mathcal{S}_{\text{Pareto}} = \{\mathbf{x} : \mathbf{x}' \not\succ \mathbf{x}, \ \forall \mathbf{x}' \in \mathcal{X}\}, \tag{1}$$

where $\mathbf{x}' \succ \mathbf{x}$ refers to $\mathbf{x}'$ dominating $\mathbf{x}$, i.e., all of the objective function values for $\mathbf{x}'$ are greater than or equal to those of $\mathbf{x}$, and at least one is greater,

$$\begin{aligned}\mathbf{x}' \succ \mathbf{x} \text{ iff } f_\bullet^l(\mathbf{x}') \geq f_\bullet^l(\mathbf{x}) \ \ \forall l \in \{1, \ldots, L\} \text{ and} \\ \exists l \in \{1, \ldots, L\} \text{ such that } f_\bullet^l(\mathbf{x}') > f_\bullet^l(\mathbf{x}).\end{aligned} \tag{2}$$

The Pareto set also induces the Pareto front, $\mathcal{F}_{\text{Pareto}} := \{\mathbf{f}_\bullet(\mathbf{x}) : \forall \mathbf{x} \in \mathcal{S}_{\text{Pareto}}\}$, which is the image of the Pareto set outcomes in $\mathbb{R}^L$.

For data collected $\mathcal{D}_N = \{(\mathbf{y}_n, \mathbf{x}_n)\}_{n=1}^N$ we define the empirical Pareto set similarly to above and and a empirical Pareto set membership label, $z_n = \mathbb{1}[\mathbf{x}_n \in \tilde{\mathcal{S}}_{\text{Pareto}}]$ where $\mathbb{1}[\cdot] : \{\text{True}, \text{False}\} \to \{1, 0\}$.

2.2. Active generation

Active generation as implemented by Steinberg et al. (2025) reframes online black-box optimization as sequential learning of a conditional generative model, guided by a CPE. At each round, $t \in \{1, \ldots, T\}$ we: (1) fit a CPE (using some proper loss, $\mathcal{L}_{\text{CPE}}$), $\pi_\theta(\mathbf{x}) \approx p(z = 1|\mathbf{x})$, parameterized by θ and where $z = \mathbb{1}[\mathbf{x} \in \mathcal{S}]$ indicates membership in some desired set, $\mathcal{S}$. For example, designs fitter than the current incumbent, $\mathbf{x}_t^*$. Then (2) update the generative model $q_\phi(\mathbf{x})$, e.g. by minimizing the reverse KL divergence to the ideal conditional, $p(\mathbf{x}|z)$, or equivalently maximizing the evidence lower bound (ELBO),

$$\mathcal{L}_{\text{ELBO}}(\phi, \theta) = \mathbb{E}_{q_\phi(\mathbf{x})}\big[\log \pi_\theta(\mathbf{x})\big] - \mathbb{D}_{\text{KL}}\left[q_\phi(\mathbf{x}) \| p(\mathbf{x}|\mathcal{D}_0)\right],$$

where $p(\mathbf{x}|\mathcal{D}_0)$ is a prior over the design space. Using data $\mathcal{D}_N^z = \{(\mathbf{x}_n, z_n)\}_{n=1}^N$, active generation optimizes, $\theta_t^* \leftarrow \operatorname{argmin}_\theta \mathcal{L}_{\text{CPE}}(\theta, \mathcal{D}_N^z)$ and $\phi_t^* \leftarrow \operatorname{argmax}_\phi \mathcal{L}_{\text{ELBO}}(\phi, \theta_t^*)$, then samples from $q_{\phi_t^*}(\mathbf{x})$ are used to propose new candidates for evaluation. New labels are acquired for these candidates, the dataset is augmented, and the process is repeated until convergence. This solution to active generation is referred to as VSD (Steinberg et al., 2025). Under certain assumptions on the form of the models, this procedure has proven convergence rates to the ideal $p(\mathbf{x}|z)$.

3. Incorporating User Preferences

In multi-objective optimization (MOO), practitioners invoke *subjective preferences* to single out a subset of designs to meet application-specific requirements. Ideally, we would like not only to incorporate these subjective preferences but also to avoid retraining our active generation framework every time a new preference is given.

3.1. Preference vectors and alignment indicators

As we will see in section 4, our solution to incorporating user preferences for active generation is based on amortization. In other words, rather than estimating a model $q_\phi(\mathbf{x})$ as in VSD, we will learn a conditional model of the form $q_\phi(\mathbf{x}|\mathbf{u})$. Consequently, we introduce *preference direction vectors* $\mathbf{u} \in \mathcal{U}$ where $\mathcal{U} = \{\mathbf{u} \in \mathbb{R}^L : \|\mathbf{u}\|_2 = 1\}$, defined from observed or desired (subjective) user specified outcomes. In our experiments, we train our method using

$$\mathbf{u}_n = g(\mathbf{y}_n) := \frac{\mathbf{y}_n - \mathbf{r}}{\|\mathbf{y}_n - \mathbf{r}\|_2}. \tag{3}$$

These unit vectors capture the relative emphasis among objectives in a single geometric object. Given a trained model, a user can specify their own preferences via $\mathbf{u}_\star = g(\mathbf{y}_\star)$ and our approach will generate solutions from $q_\phi(\mathbf{x}|\mathbf{u}_\star)$. We note here that, instead of achieving uniform performance over all possible $\mathbf{u}$s, our goal is to prioritize exploration of designs in certain regions of the Pareto set, and not others. Importantly, our generative model needs to enforce that generated samples respect a user's desired trade-off. Therefore, we define an *alignment indicator*, $a \in \{0, 1\}$, that labels each $(\mathbf{x}, \mathbf{u})$ pair as 'aligned' if it achieves correct projection onto the preference direction.

Preference directions generalize (convex) scalarization weights to a generative setting: any $\boldsymbol{\lambda}$ can be mapped to a unit-norm vector $\mathbf{u} = \boldsymbol{\lambda}/\|\boldsymbol{\lambda}\|_2$, and conversely each $\mathbf{u}$ induces a unique normalized weight. By conditioning on $(\mathbf{u}, a)$ rather than $\boldsymbol{\lambda}$ alone, our generative Pareto-set model becomes both more flexible (no retraining for new trade-offs) and more faithful to non-dominance structure. We visualize these preference direction vectors in Figure 1a.

4. Amortized Active Generation of Pareto Sets

We now have all the components to describe our amortized active generation framework that learns to generate (approximate) solutions in the Pareto set, conditioned on user preferences. We call our method active generation of Pareto sets (A-GPS), and it begins by generalizing the active generation objective in Steinberg et al. (2025). That is, for each round, t, we minimize the reverse KL divergence between the generative model $q_\phi(\mathbf{x}|\mathbf{u})$ and an underlying (unobserved) true model $p(\mathbf{x}|\mathbf{u}, z, a)$,

$$\phi_t^* = \operatorname*{argmin}_{\phi} \mathbb{D}_{\mathrm{KL}}[q_\phi(\mathbf{x}|\mathbf{u})p(\mathbf{u}|z) \| p(\mathbf{x}|\mathbf{u}, z, a)p(\mathbf{u}|z)] ,$$
$$= \operatorname*{argmin}_{\phi} \mathbb{E}_{p(\mathbf{u}|z)}\big[\mathbb{D}_{\mathrm{KL}}[q_\phi(\mathbf{x}|\mathbf{u}) \| p(\mathbf{x}|\mathbf{u}, z, a)]\big] . \quad (4)$$

The inclusion of $p(\mathbf{u}|z)$ rewards learning an *amortized* generative model, $q_\phi(\mathbf{x}|\mathbf{u})$, over the distribution of the relevant preference directions. Naturally we cannot evaluate $p(\mathbf{x}|\mathbf{u}, z, a)$ directly, and so we appeal to Bayes' rule,

$$p(\mathbf{x}|\mathbf{u}, z, a) = \frac{1}{Z} p(z|\mathbf{x}, \mathbf{u})p(a|\mathbf{x}, \mathbf{u})p(\mathbf{x}|\mathbf{u}). \quad (5)$$

Here we have assumed conditional independence between z and a given $\mathbf{x}$ and $\mathbf{u}$, and since $Z = p(z, a|\mathbf{u})$ is a constant w.r.t. $\mathbf{x}$, we will omit it from our objective. We make a further simplifying assumption that a-priori $p(\mathbf{x}|\mathbf{u}) = p(\mathbf{x}|\mathcal{D}_0)$, and then we rely on the likelihood guidance terms, $p(z|\mathbf{x}, \mathbf{u})p(a|\mathbf{x}, \mathbf{u})$, to capture the joint relationship between $(\mathbf{x}, \mathbf{u})$ in the variational posterior. We justify this decision by noting that the relationship, $\mathbf{x}|\mathbf{u}$, may be difficult to reason about a-priori, and requiring such a prior would then preclude the use of pre-trained models for $p(\mathbf{x}|\mathcal{D}_0)$. Putting this all together results in the following equivalent amortized ELBO objective, $\phi_t^* = \operatorname{argmax}_\phi \mathcal{L}_{\text{A-ELBO}}(\phi)$ where,

$$\mathcal{L}_{\text{A-ELBO}}(\phi) = \mathbb{E}_{\underbrace{p(\mathbf{u}|z)}_{\text{Direction dist.}}}\big[\mathbb{E}_{q_\phi(\mathbf{x}|\mathbf{u})}\big[\log \underbrace{p(z|\mathbf{x}, \mathbf{u})}_{\text{Pareto CPE}}$$
$$+ \log \underbrace{p(a|\mathbf{x}, \mathbf{u})}_{\text{Align. CPE}} - \mathbb{D}_{\mathrm{KL}}[q_\phi(\mathbf{x}|\mathbf{u}) \| p(\mathbf{x}|\mathcal{D}_0)]\big]\big] . \quad (6)$$

We will now discuss how we estimate each of these components in turn, leading to the A-GPS algorithm presented in Appendix C.

4.1. Estimating A-GPS's component distributions

Preference direction distribution, $p(\mathbf{u}|z)$. Since we observe $\mathbf{u}_n$, we can approximate empirically $p(\mathbf{u}|z) \approx |\tilde{\mathcal{S}}_{\text{Pareto}}|^{-1} \sum_{n=1}^{N} z_n \mathbb{1}[\mathbf{u} = \mathbf{u}_n]$. However, we find that this can occasionally lead to an overly exploitative strategy for black-box optimization. Instead, we use maximum likelihood to learn a parameterized estimator $q_\gamma(\mathbf{u}) \approx p(\mathbf{u}|z)$, with data $\mathcal{D}_N^z = \{(\mathbf{x}_n, \mathbf{u}_n, z_n)\}_{n=1}^N$, $\gamma_t^* = \operatorname{argmin}_\gamma \mathcal{L}_{\text{Pref}}(\gamma, \mathcal{D}_N^z)$, where

$$\mathcal{L}_{\text{Pref}}(\gamma, \mathcal{D}_N^z) = -\frac{1}{|\tilde{\mathcal{S}}_{\text{Pareto}}|} \sum_{n=1}^{N} z_n \log q_\gamma(\mathbf{u}_n). \quad (7)$$

Examples of appropriate parametric forms are von Mises-Fisher distributions, power spherical distributions (De Cao & Aziz, 2020) or normalizing flows (Rezende et al., 2020). We find Normal distributions, or mixtures, normalized to the unit sphere are more numerically stable than some of the specialized spherical distributions, see Sec. H.1 for more detail.

Pareto CPE, $p(z|\mathbf{x}, \mathbf{u})$. As per the original VSD, we define a CPE to directly discriminate over the solution set, which in this case is $\tilde{\mathcal{S}}_{\text{Pareto}}$. With data, $\mathcal{D}_N^{\mathbf{z}}$, we use the log-loss,

$$\mathcal{L}_{\text{CPE}}^z(\theta, \mathcal{D}_N^z) = -\frac{1}{N} \sum_{n=1}^{N} z_n \log \pi_\theta(\mathbf{x}_n, \mathbf{u}_n)$$
$$+ (1 - z_n) \log(1 - \pi_\theta(\mathbf{x}_n, \mathbf{u}_n)), \quad (8)$$

where $\pi_\theta(\mathbf{x}, \mathbf{u})$ is a discriminative model parameterized by θ, e.g. a neural network. To acquire labels z_n defining Pareto set membership we could make use of fast dominance checking (Kung et al., 1975). In practice we find the dominance checker in Balandat et al. (2020) sufficient for our purposes.

Preference alignment CPE, $p(a|\mathbf{x}, \mathbf{u})$. Since we do not wish to require a strong prior, $p(\mathbf{x}|\mathbf{u})$, to be our only source of preference alignment information, we instead explicitly reward alignment in our conditional generative model by using a CPE guide. We create contrastive data for training this guide, $\mathcal{D}_N^a = \{(a_n = 1, \mathbf{x}_n, \mathbf{u}_n)\}_{n=1}^N \cup \{(\tilde{a}_n = 0, \mathbf{x}_n, \mathbf{u}_{\rho(n)}\}_{n=1}^N$ where the second set are purposefully misaligned by a random permutation, $\rho : \mathbb{N} \to \mathbb{N}$. This results in the log-loss,

$$\mathcal{L}_{\text{CPE}}^a(\psi, \mathcal{D}_N^a) = -\frac{1}{N} \sum_{n=1}^{N} \log \pi_\psi(\mathbf{x}_n, \mathbf{u}_n)$$
$$- \frac{1}{N} \sum_{n=1}^{N} \log(1 - \pi_\psi(\mathbf{x}_n, \mathbf{u}_{\rho(n)})), \quad (9)$$

where $\pi_\psi(\mathbf{x}, \mathbf{u})$ is our CPE parameterized by ψ.

4.2. Learning A-GPS's variational distribution

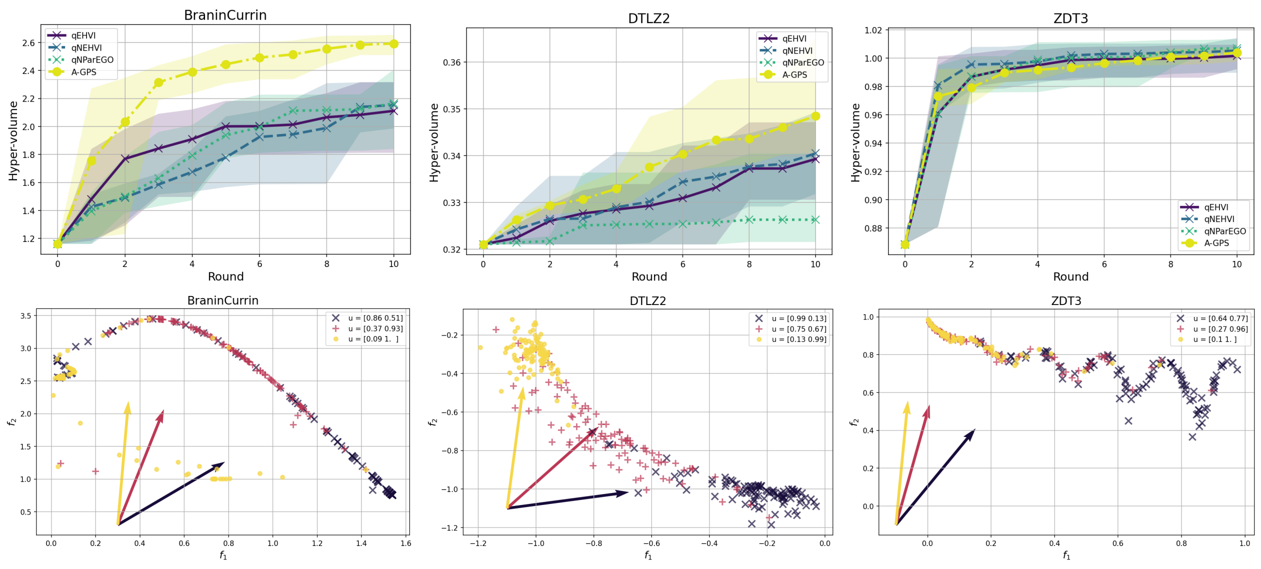
To learn $q_\phi(\mathbf{x}|\mathbf{u})$, we can now re-write our amortized ELBO, Equation 6, in terms of these estimated quantities,

$$\mathcal{L}_{\text{A-ELBO}}(\phi, \theta, \psi, \gamma) = \mathbb{E}_{q_\gamma(\mathbf{u})}\big[\mathbb{E}_{q_\phi(\mathbf{x}|\mathbf{u})}\big[\log \pi_\theta(\mathbf{x}, \mathbf{u})$$
$$+ \log \pi_\psi(\mathbf{x}, \mathbf{u})\big] - \mathbb{D}_{\mathrm{KL}}[q_\phi(\mathbf{x}|\mathbf{u}) \| p(\mathbf{x}|\mathcal{D}_0)]\big] . \quad (10)$$

We find that using 'on-policy' gradient estimation methods, such as REINFORCE (Williams, 1992; Mohamed et al., 2020), are very slow when we have complex variational distribution forms, $q_\phi(\mathbf{x}|\mathbf{u})$, such as long short-term memory (LSTM) recurrent neural networks (RNNs) or decoder-only Transformers. This is because we have to set a low learning rate as the variance of this estimator can induce exploding gradients for long sequences, and new samples have to be drawn from the variational distribution every iteration of stochastic gradient descent (SGD). So instead we use an 'off-policy' gradient estimator with importance weights to emulate the on-policy estimator, see Appendix B for details.

4.3. Generating Pareto-set candidates for evaluation

In round t, we are free to choose $\mathbf{u}_{bt}$ based on preferences, or if we do not have specific preferences to incorporate into the query we could sample $\{\mathbf{u}_{bt}\}_{b=1}^B \sim$



(a) Negative Branin-Currin (b) DTLZ2 (c) ZDT3

Figure 2: Experimental results on three test functions commonly used in the MOBO literature. The top row reports hyper-volume indicator (HVI) per round, the bottom row demonstrates amortized preference conditioning.

$\prod_{b=1}^{B} q_{\gamma_t^*}(\mathbf{u})$. To recommend candidates for black-box evaluation we sample a set of B designs from our search distribution, $\{\mathbf{x}_{bt}\}_{b=1}^{B} \sim \prod_{b=1}^{B} q_{\phi_t^*}(\mathbf{x}|\mathbf{u}_{bt})$, where $\phi_t^* = \text{argmax}_\phi \, \mathcal{L}_{\text{A-ELBO}}(\phi, \theta_t^*, \psi_t^*, \gamma_t^*)$. We present the full online optimization algorithm in Appendix C.

5. Experiments

We evaluate A-GPS on a number of benchmarks and compare it to some popular baselines. We report results on synthetic data here and on real applications in Appendix F, with all experimental details in Appendix I.

Synthetic test functions: A detailed description of these functions and/or their Pareto-front geometries can be found in Appendix E and Zhang & Li (2007); Belakaria et al. (2019). Although A-GPS was originally developed for discrete/mixed spaces, we directly apply it to these continuous domains using a conditional Gaussian generative model, $q_\phi(\mathbf{x}|\mathbf{u}) = \mathcal{N}\big(\mathbf{x}|\boldsymbol{\mu}(\mathbf{u}), \boldsymbol{\sigma}^2(\mathbf{u})\big)$ with mean and variance, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$, given by a neural network (NN). We find that it nonetheless achieves strong performance.

The top row of Figure 2 reports mean hyper-volume versus optimization round (with shaded bands indicating ± 1 std from 5 runs). All methods use 64 training points, and then recommend 5 candidates per round. On Branin–Currin, A-GPS (yellow) rapidly outpaces the Gaussian process (GP)–based baselines (qNEHVI (Daulton et al., 2021), qEHVI and qNParEGO (Daulton et al., 2020)), achieving higher front coverage with fewer evaluations. On the non-convex DTLZ2 and more fragmented ZDT3 landscapes, all methods converge to similar final hyper-volumes (excepting qNParEGO), but A-GPS nevertheless matches their sample efficiency despite its non-GP, generative formulation. The bottom row of Figure 2 illustrates preference conditioning: each panel plots the sampled Pareto front (dots) from $q_\phi(\mathbf{x}|\mathbf{u}_*)$ colored by three representative preference directions $\mathbf{u}_*$ (see Sec. E.1). In most cases our generated samples (dots) correspond to their preference direction vectors (arrows), except for the yellow arrow in the Branin-Currin experiment. This preference direction points just outside $\tilde{\mathcal{S}}_{\text{Pareto}}$, and so illustrates how our amortized model behaves under unrealistic preference requests. Though typically we can rectify this behavior by marginalizing over all $p(\mathbf{u})$ instead of $p(\mathbf{u}|z)$, if we are prepared to reduce sample efficiency. Overall, these results show that A-GPS not only excels at front approximation but also supports flexible, a-posteriori preference conditioning across a variety of continuous landscapes.

Discussion and Conclusion We have developed active generation of Pareto sets (A-GPS), a method learns a preference-conditioned generative model $q_\phi(\mathbf{x}|\mathbf{u})$ of the Pareto set using flexible architectures, including LSTMs and transformers. Unlike diffusion-based approaches, A-GPS is modular and adaptable. Empirical results on synthetic benchmarks and protein design tasks show strong performance. We expect this framework to enable future extensions with diverse generative architectures for large-scale multi-objective optimization.

References

Ament, S., Daulton, S., Eriksson, D., Balandat, M., and Bakshy, E. Unexpected improvements to expected improvement for Bayesian optimization. *Advances in Neural Information Processing Systems*, 36:20577–20612, 2023.

Annadani, Y., Belakaria, S., Ermon, S., Bauer, S., and Engelhardt, B. E. Preference-guided diffusion for multi-objective offline optimization. *arXiv preprint arXiv:2503.17299*, 2025.

Balandat, M., Karrer, B., Jiang, D. R., Daulton, S., Letham, B., Wilson, A. G., and Bakshy, E. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in Neural Information Processing Systems 33*, 2020.

Belakaria, S., Deshwal, A., and Doppa, J. R. Max-value entropy search for multi-objective bayesian optimization. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Blaabjerg, L. M., Kassem, M. M., Good, L. L., Jonsson, N., Cagiada, M., Johansson, K. E., Boomsma, W., Stein, A., and Lindorff-Larsen, K. Rapid protein stability prediction using deep learning representations. *eLife*, 12:e82593, May 2023. ISSN 2050-084X. doi: 10.7554/eLife.82593.

Brookes, D., Park, H., and Listgarten, J. Conditioning by adaptive sampling for robust design. In *International conference on machine learning*, pp. 773–782. PMLR, 2019.

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. In *The Fourth International Conference on Learning Representations*, 2016.

Daulton, S., Balandat, M., and Bakshy, E. Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization. *Advances in Neural Information Processing Systems*, 33:9851–9864, 2020.

Daulton, S., Balandat, M., and Bakshy, E. Parallel Bayesian optimization of multiple noisy objectives with expected hypervolume improvement. *Advances in Neural Information Processing Systems*, 34:2187–2200, 2021.

De Ath, G., Chugh, T., and Rahat, A. A. Mbore: multi-objective Bayesian optimisation by density-ratio estimation. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 776–785, 2022.

De Cao, N. and Aziz, W. The power spherical distribution. *arXiv preprint arXiv:2006.04437*, 2020.

Dhariwal, P. and Nichol, A. Q. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

Ferruz, N., Schmidt, S., and Höcker, B. Protgpt2 is a deep unsupervised language model for protein design. *Nature communications*, 13(1):4348, 2022.

González-Duque, M., Bartels, S., and Michael, R. poli: a libary of discrete sequence objectives, January 2024. URL `https://github.com/MachineLearningLifeScience/poli`.

Gruver, N., Stanton, S., Frey, N., Rudner, T. G., Hotzel, I., Lafrance-Vanasse, J., Rajpal, A., Cho, K., and Wilson, A. G. Protein design with guided discrete diffusion. *Advances in neural information processing systems (NeurIPS)*, 36, 2023.

Hernández-Lobato, D., Hernandez-Lobato, J., Shah, A., and Adams, R. Predictive entropy search for multi-objective bayesian optimization. In *International conference on machine learning*, pp. 1492–1501. PMLR, 2016.

Jain, M., Raparthy, S. C., Hernández-Garcıa, A., Rector-Brooks, J., Bengio, Y., Miret, S., and Bengio, E. Multi-objective gflownets. In *International conference on machine learning*, pp. 14631–14653. PMLR, 2023.

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

Klarner, L., Rudner, T. G. J., Morris, G. M., Deane, C. M., and Teh, Y. W. Context-guided diffusion for out-of-distribution molecular and protein design. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Knowles, J. Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE transactions on evolutionary computation*, 10(1):50–66, 2006.

Kung, H.-T., Luccio, F., and Preparata, F. P. On finding the maxima of a set of vectors. *Journal of the ACM (JACM)*, 22(4):469–476, 1975.

Lin, X., Yang, Z., Zhang, X., and Zhang, Q. Pareto set learning for expensive multi-objective optimization. *Advances in neural information processing systems*, 35: 19231–19247, 2022.

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020.

Paria, B., Kandasamy, K., and Póczos, B. A flexible framework for multi-objective Bayesian optimization using random scalarizations. In *Uncertainty in Artificial Intelligence*, pp. 766–776. PMLR, 2020.

Perez, E., Strub, F., De Vries, H., Dumoulin, V., and Courville, A. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Precup, D., Sutton, R. S., and Singh, S. Eligibility traces for off-policy policy evaluation. In *ICML*, volume 2000, pp. 759–766. Citeseer, 2000.

Rezende, D. J., Papamakarios, G., Racaniere, S., Albergo, M., Kanwar, G., Shanahan, P., and Cranmer, K. Normalizing flows on tori and spheres. In *International Conference on Machine Learning*, pp. 8083–8092. PMLR, 2020.

Rubinstein, R. Y. and Kroese, D. P. *Simulation and the Monte Carlo method*. John Wiley & Sons, 2016.

Stanton, S., Maddox, W., Gruver, N., Maffettone, P., Delaney, E., Greenside, P., and Wilson, A. G. Accelerating Bayesian optimization for biological sequence design with denoising autoencoders. In *International Conference on Machine Learning*, pp. 20459–20478. PMLR, 2022.

Stanton, S., Alberstein, R., Frey, N., Watkins, A., and Cho, K. Closed-form test functions for biophysical sequence optimization algorithms. *arXiv preprint arXiv:2407.00236*, 2024.

Steinberg, D. M., Oliveira, R., Ong, C. S., and Bonilla, E. V. Variational search distributions. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Wang, C., Uehara, M., He, Y., Wang, A., Lal, A., Jaakkola, T., Levine, S., Regev, A., Hanchen, and Biancalani, T. Fine-tuning discrete diffusion models via reward optimization with applications to DNA and protein design. In *The Thirteenth International Conference on Learning Representations*, 2025.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Yang, K., Emmerich, M., Deutz, A., and Bäck, T. Efficient computation of expected hypervolume improvement using box decomposition algorithms. *Journal of Global Optimization*, 75:3–34, 2019.

Yao, Y., Pan, Y., Li, J., Tsang, I., and Yao, X. Proud: Pareto-guided diffusion model for multi-objective generation. *Machine Learning*, 113(9):6511–6538, 2024.

Yuan, Y., Chen, C., Pal, C., and Liu, X. Paretoflow: Guided flows in multi-objective optimization. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Zhang, Q. and Li, H. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007. doi: 10.1109/TEVC.2007.892759.

Algorithm 1 A-GPS optimization loop.

```
Require: Initial dataset $\mathcal{D}_N$, black-box $\mathbf{f}_\bullet$, prior $p(\mathbf{x}|\mathcal{D}_0)$, CPEs $\pi_\theta(\mathbf{x},\mathbf{u})$ and $\pi_\psi(\mathbf{x},\mathbf{u})$, variational families $q_\gamma(\mathbf{u})$ and $q_\phi(\mathbf{x}|\mathbf{u})$,
    alignment threshold $\tau$, budget $T$ and $B$.
 1: function FITMODELS($\mathcal{D}_N$)
 2:     $\mathcal{D}^z_N \leftarrow \{(z_n, \mathbf{x}_n, \mathbf{u}_n)\}^N_{n=1}$,  where $z_n = \mathbb{1}[\mathbf{x}_n \in \tilde{\mathcal{S}}_{\text{Pareto}}]$ and $\mathbf{u}_n = (\mathbf{y}_n - \mathbf{r})/\|\mathbf{y}_n - \mathbf{r}\|$
 3:     $\mathcal{D}^a_N \leftarrow \{(a_n = 1, \mathbf{x}_n, \mathbf{u}_n)\}^N_{n=1} \cup \{(a_n = 0, \mathbf{x}_n, \mathbf{u}_{\rho(n)})\}^N_{n=1}$
 4:     $\gamma^* \leftarrow \text{argmin}_\gamma \mathcal{L}_{\text{Pref}}(\gamma, \mathcal{D}^z_N)$
 5:     $\theta^* \leftarrow \text{argmin}_\theta \mathcal{L}^z_{\text{CPE}}(\theta, \mathcal{D}^z_N)$
 6:     $\psi^* \leftarrow \text{argmin}_\psi \mathcal{L}^a_{\text{CPE}}(\psi, \mathcal{D}^a_N)$
 7:     $\phi^* \leftarrow \text{argmax}_\phi \mathcal{L}_{\text{A-ELBO}}(\phi, \theta^*, \psi^*, \gamma^*)$
 8:     return $\phi^*, \theta^*, \psi^*, \gamma^*$
 9: for round $t \in \{1, \ldots, T\}$ do
10:     $\phi^*_t, \theta^*_t, \psi^*_t, \gamma^*_t \leftarrow$ FITMODELS($\mathcal{D}_N$)
11:     $\{\mathbf{u}_{bt}\}^B_{b=1} \leftarrow$ sample $q_{\gamma^*_t}(\mathbf{u})$
12:     $\{\mathbf{x}_{bt}\}^B_{b=1} \leftarrow$ sample $q_{\phi^*_t}(\mathbf{x}|\mathbf{u}_{bt})$  $\forall b \in \{1, \ldots, B\}$
13:     $\{\mathbf{y}_{bt}\}^B_{b=1} \leftarrow \{\mathbf{f}_\bullet(\mathbf{x}_{bt}) + \boldsymbol{\epsilon}_{bt}\}^B_{b=1}$
14:     $\mathcal{D}_N \leftarrow \mathcal{D}_N \cup \{(\mathbf{x}_{bt}, \mathbf{y}_{bt})\}^B_{b=1}$
15: $\phi^*, \theta^*, \psi^*, \gamma^* \leftarrow$ FITMODELS($\mathcal{D}_N$)
16: return $\phi^*, \theta^*, \psi^*, \gamma^*$
```

A. Limitations and Broader Impacts

A limitation with A-GPS, and one that it shares with many MOBO and MOG methods, is that it can be hard to specify algorithm hyper-parameters a-priori—before new data has been acquired—and the settings of these hyper-parameters can effect real-world performance. We are mindful of this in our implementation and design of A-GPS in that it comprises components that can be independently trained and validated meaningfully on the training data at hand (e.g. the CPEs, preference distribution and prior if appropriate).

We attempt to give an honest accounting of our method, and do make an attempt to show some of its failure modes and limitations that we have encountered, as well as its strengths. This work is motivated by applications that aim to improve societal sustainability, for example, through the engineering of enzymes to help control harmful waste. However, as with many technologies, it carries the risk of misuse by malicious actors. We, the authors, explicitly disavow and do not condone such uses.

B. Gradients

$$\nabla_\phi \mathcal{L}_{\text{A-ELBO}}(\phi, \theta, \psi, \gamma) = \mathbb{E}_{q_{\phi'}(\mathbf{x}|\mathbf{u})q_\gamma(\mathbf{u})}\Big[w(\mathbf{x},\mathbf{u}) \cdot \left(\log \pi_\theta(\mathbf{x},\mathbf{u}) + \log \pi_\psi(\mathbf{x},\mathbf{u}) - \log \frac{q_\phi(\mathbf{x}|\mathbf{u})}{p(\mathbf{x}|\mathcal{D}_0)}\right) \nabla_\phi \log q_\phi(\mathbf{x}|\mathbf{u})\Big]. \quad (11)$$

Here $w(\mathbf{x},\mathbf{u}) = q_\phi(\mathbf{x}|\mathbf{u})/q_{\phi'}(\mathbf{x}|\mathbf{u})$, or some normalized version (Rubinstein & Kroese, 2016), are the importance weights (Precup et al., 2000; Burda et al., 2016). Now we use S samples from $\mathbf{x}^{(s)} \sim q_{\phi'}(\mathbf{x}|\mathbf{u}^{(s)})$, to approximate the expectation in Equation 11, where we sample S once each round (t) and when the effective sample size drops below a predetermined threshold ($0.5S$). We typically choose $\phi' = \phi^*_{t-1}$, or if we choose $\phi' = \phi$ we recover on-policy gradients. We then use these gradients with an appropriate SGD algorithm, such as Adam (Kingma & Ba, 2014), to optimize for ϕ^*_t.

C. The A-GPS Algorithm

The A-GPS algorithm is described in Algorithm 1.

D. Related Work

Our work sits at the intersection of online black-box optimization, generative modeling, and user-guided multi-objective search. We organize existing methods along three primary dimensions: whether they operate online or offline, whether they directly optimize acquisition functions or learn conditional generative models for optimization, if they use guidance or

Table 1: Comparison of recent MOG and related techniques. '✓' means the method has the feature, '✗' the method lacks the feature and '–' the method can be extended to incorporate the feature. 'Modular' refers to the non-specific nature of the variational distribution used by conditioning by adaptive sampling (CbAS), VSD and A-GPS, i.e., it can be chosen based on the task.

Method	Designed for MOO	Online black-box optimization (BBO)	Amortized preference conditioning	Non-convex Pareto front	Discrete/mixed $\mathcal{X}$	Generative pref. model, $q_\gamma(\mathbf{u})$	Generative obs. model, $q_\phi(\boldsymbol{x})$	Guide
LaMBO (Stanton et al., 2022)	✓	✓	✗	✓	✓	✗	Masked LM	nEHVI
LaMBO-2 (Gruver et al., 2023)	✓	✓	✗	✓	✓	✗	Diffusion	nEHVI
Pareto Set Learning (PSL) (Lin et al., 2022)	✓	✓	✓	✓	✗	✗	Deterministic MLP	Scalarization
GFlowNets (Jain et al., 2023)	✓	✓	✓	✓	✓	✗	GFlowNets	Scalarization
ParetoFlow (Yuan et al., 2025)	✓	✗	✗	✗	✓	–	Diffusion	Scalarization
PROUD (Yao et al., 2024)	✓	✗	✗	✓	–	✗	Diffusion	Multiple grad. desc.
Preference Guided Diffusion (Annadani et al., 2025)	✓	✗	✗	✓	✗	✗	Diffusion	Preference CPE
CbAS (Brookes et al., 2019)	–	–	✗	✓	✓	✗	Modular	Dominance CPE
VSD (Steinberg et al., 2025)	–	✓	✗	✓	✓	✗	Modular	Dominance CPE
A-GPS (ours)	✓	✓	✓	✓	✓	✓	Modular	Dominance CPE

amortization for generation.

Online vs. Offline. Traditional MOBO methods, such as hypervolume-based acquisitions (EHVI, noisy expected hypervolume improvement (nEHVI), and their variants), and entropy search (Yang et al., 2019; Daulton et al., 2020; 2021; Hernández-Lobato et al., 2016) and scalarization methods (Knowles, 2006; Zhang & Li, 2007; Paria et al., 2020; De Ath et al., 2022), operate online by sequentially querying the black-box using acquisition rules that balance exploration and exploitation. In contrast, offline MOG approaches like ParetoFlow and guided diffusion frameworks (Yuan et al., 2025; Yao et al., 2024; Annadani et al., 2025) train generative models from a fixed dataset of evaluated designs, without further oracle queries. While these offline methods can leverage rich generative priors, they have not been designed to adapt to new information.

Generative Models vs. Acquisition Optimization. Recent advances in "active generation" recast black-box optimization as fitting conditional generative models to high-value regions, guided by predictors and/or acquisition functions. Methods like VSD, GFlowNets, and diffusion-based solvers (Steinberg et al., 2025; Jain et al., 2023; Dhariwal & Nichol, 2021; Gruver et al., 2023) show that generative search can match or exceed traditional direct acquisition function optimization, particularly in large search spaces. However, existing generative frameworks often need re-training to integrate subjective preferences. Similarly, all direct acquisition optimization methods require additional optimization runs to incorporate new preferences. An exception is Pareto set learning (Lin et al., 2022), which learns a neural-net, that maps from scalarization weights to designs.

Guidance vs. Amortization. Or inference-time vs. re-training/fine-tuning based search. Guided generation methods, such as those based on guided diffusion and flow matching (Gruver et al., 2023; Yao et al., 2024; Yuan et al., 2025; Annadani et al., 2025) use a pre-trained generative model, from which samples are then *guided* at inference time such that they are generated from a conditional generative model, leaving the original generative model unchanged. It has been noted in (Klarner et al., 2024) that guided methods, though computationally efficient, may suffer from co-variate shift issues when guiding too far from the support of the pretrained model. Conversely, amortized methods such as (Steinberg et al., 2025; Wang et al., 2025) explicitly re-train or fine-tune the generative model to condition it, thereby circumventing these co-variate shift issues at the cost of more computation, but allowing for less constrained exploration in online scenarios.

Our A-GPS approach unifies these dimensions: it learns a conditional generative model online, bypasses explicit acquisition optimization, and uses amortization to avoid co-variate shift. Table 1 compares key features across representative MOG methods.

E. Synthetic Test Functions

To demonstrate the efficacy of A-GPS on classical multi-objective landscapes, we evaluate on three two-objective ($L = 2$) continuous ($\mathbf{x} \in \mathbb{R}^D$) problems commonly used for MOBO (Zhang & Li, 2007; Belakaria et al., 2019; Balandat et al., 2020):

Branin-Currin ($D = 2$): We optimize the negative Branin-Currin convex pair. We found the negative of this function has a more interesting Pareto front while remaining a challenging MOBO task.
DTLZ2 ($D = 3$): A smooth, spherical front in the negative orthant, which tests an algorithm's ability to approximate non-convex curved manifolds in higher dimensions.
ZDT3 ($D = 6$): A discontinuous non-convex front comprised of several disconnected segments, which stresses an optimizer's capacity for both exploration and front-segment coverage.

E.1. Preference directions

These $\mathbf{u}_*$ were chosen by,

$$\mathbf{y}_* \in \{[Q_{\tilde{\mathcal{F}}^1_{\text{Pareto}}}(0.9), Q_{\tilde{\mathcal{F}}^2_{\text{Pareto}}}(0.1)], [\text{Av}(\tilde{\mathcal{F}}^1_{\text{Pareto}}), \text{Av}(\tilde{\mathcal{F}}^2_{\text{Pareto}})], [Q_{\tilde{\mathcal{F}}^1_{\text{Pareto}}}(0.1), Q_{\tilde{\mathcal{F}}^2_{\text{Pareto}}}(0.9)]\}, \tag{12}$$

where Q is an empirical quantile function and Av denotes the set mean of each dimension, l, of the empirical Pareto front, $\tilde{\mathcal{F}}^l_{\text{Pareto}}$. We then use Equation 3 to convert these into $\mathbf{u}_*$ with a problem specific reference point $\mathbf{r}$.

F. Experiments on Real Data

F.1. Ehrlich vs. Naturalness

We now evaluate A-GPS on a challenging two-objective 'peptide' design task that couples the Ehrlich synthetic landscape (Stanton et al., 2024) with a ProtGPT2 (Ferruz et al., 2022) 'naturalness' score. The Ehrlich function has been designed to emulate key aspects of protein fitness; it maps each discrete sequence to a scalar by embedding combinatorial 'motif' interactions in a highly rugged and multi-modal terrain. Its epistatic peaks and valleys capture the statistical complexity of peptide design, yet it is entirely artificial, bearing no biochemical or evolutionary validity. In stark contrast, ProtGPT2's log-probability reflects genuine amino-acid patterns learned from 250 million real proteins. We use protein sequences $\mathcal{X} = \mathcal{V}^M$ where $|\mathcal{V}| = 20$ and $M \in \{15, 32, 64\}$. The two objective are,

$$f_\bullet^1(\mathbf{x}) = \text{Ehrlich}(\mathbf{x}), \qquad f_\bullet^2(\mathbf{x}) = 15^{-1}(\log p_{\text{ProtGPT2}}(\mathbf{x}) + 15), \tag{13}$$

where we scale ProtGPT2's score to be in a comparable range as the Ehrlich function for sensible scaling of HVI. We compare against CbAS (Brookes et al., 2019) and VSD (Steinberg et al., 2025), which we have modified to use the same Pareto CPE as A-GPS, and against the guided diffusion LaMBO-2 method of (Gruver et al., 2023), which is formulated for discrete MOBO tasks. CbAS and VSD use a causal transformer architecture and a LSTM for their $q_\phi(\mathbf{x})$. A-GPS uses the same base architectures, but embeds $\mathbf{u}$ for it's prefix token to condition the transformer, and uses FiLM (Perez et al., 2018) for conditioning the LSTM. The same (unconditional) architectures are used as priors, which are trained on the initial sequences using maximum likelihood. We also compare to a naïve random mutation method that can randomly mutate 3 amino acids per sequence per round.

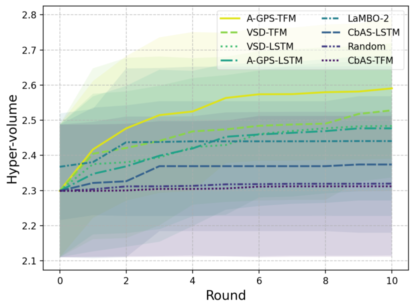
Results are reported in Figure 3 for 10 runs from random starting conditions (bands indicating ± 1 std). All methods are given 128 training samples, and the recommend batches of size 32 per round. We use the `poli` and `poli-baselines` libraries for running the benchmarks and LaMBO-2 baseline (González-Duque et al., 2024). A-GPS performs favorably compared to most other methods, though the challenging epistatic and discrete nature of the Ehrlich function results in a large deviation between runs.

F.2. RaSP vs. Naturalness

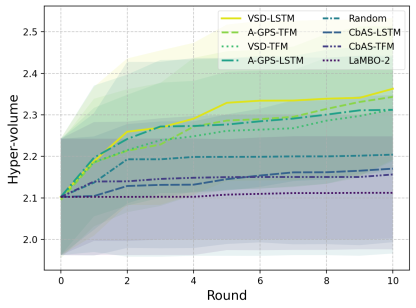
For the final experiment we wish to optimize rapid stability predictions (RaSP) (Blaabjerg et al., 2023) against the same ProtGPT2-based naturalness score we used in the previous experiment. RaSP is a deep-learning model trained to predict the thermodynamic stability of protein structures, which is of particular importance in protein engineering if, e.g. we are designing proteins for an industrial application. The task is to engineer the DsRED.M1 protein from (Stanton et al., 2022), which is a longer sequence with with $M = 217$, to maximize its stability and naturalness scores defined by

$$f_\bullet^1(\mathbf{x}) = 100^{-1}(\text{RaSP}(\mathbf{x}) + 100), \qquad f_\bullet^2(\mathbf{x}) = 15^{-1}(\log p_{\text{ProtGPT2}}(\mathbf{x}) + 15), \tag{14}$$

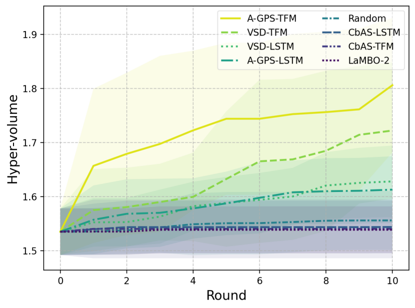
again using scaling to make the scores more comparable with HVI. The experimental setting is the same as for the $M = 64$ Ehrlich vs. Naturalness setting, and we report results in Figure 4. Interestingly in this experiment A-GPS-LSTM would



(a) $M = 15$

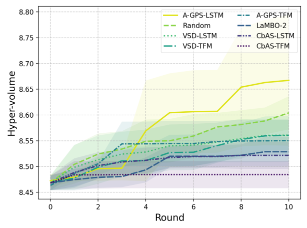


(b) $M = 32$

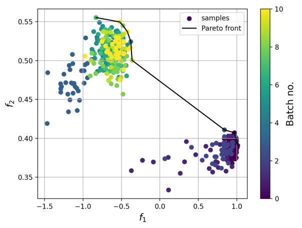


(c) $M = 64$

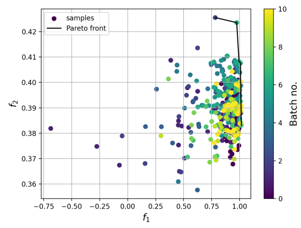
Figure 3: Ehrlich function vs. ProtGPT2 naturalness score. A-GPS, VSD and CbAS with different variational distributions; LSTM and transformer (TFM), compared against a random mutation and LaMBO-2 baselines.



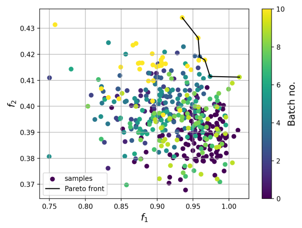
(a) Hyper-volume



(b) A-GPS-LSTM



(c) VSD-Transformer



(d) Random Mutations

Figure 4: Rapid stability predictions (RaSP) vs. ProtGPT2 naturalness score. (b) to (d) visualize the sampled $\mathbf{y}_n$ as well as the estimated $\tilde{\mathcal{F}}_{\text{Pareto}}$. Samples are colored by batch round number, t.

find a disconnected section of the Pareto front about half the time that no other method would find; we have visualized this in Figure 6b. Unfortunately no other method seemed to improve HVI over the random method, even though they were consistently more efficient at sampling the estimated Pareto set, see Figure 6c and Figure 6d.

G. Extra Experiments

G.1. Missing LaMBO-2 comparisons

The version of LaMBO-2 (Gruver et al., 2023) implemented in `poli-baselines` (González-Duque et al., 2024) is not natively a MOO solver, even though the original method is when using an EHVI acquisition function.

As a consequence, we were not able to achieve a performance adequate to represent the original work's capabilities in time for the main paper deadline. Subsequently, we been able to improve the `poli-baseline` implementation, and so re-present the results from Figure 3 and Figure 4 here in Figure 5 and Figure 6.

H. Architectural Details

H.1. Preference direction distributions

In all the experiments we use a mixture of isotropic Normal distributions where the samples have been constrained to the unit norm,

$$q_\gamma(\mathbf{u}) = \sum_{k=1}^{K} w_k \mathcal{N}_{\|\mathbf{u}\|}(\mathbf{u}|\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k^2), \tag{15}$$

and $\gamma = \{(\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k)\}_{k=1}^K$. Typically, we find $K = 5$ is sufficient. We learn this via maximum likelihood as per Equation 7, but we add an extra regularisation term: $-\frac{1}{K}\sum_{k=1}^{K}(\|\boldsymbol{\mu}_k\| - 1)^2$ so the magnitude of the mixture means is controlled (and does not decrease to 0 or increase to $\pm\infty$). We have compared this to von Mises distributions, and find it more numerically stable, we also find no tangible benefit using more complex spherical normalizing flow representations (Rezende et al., 2020). Furthermore, we find that the performance is similar to, if not slightly superior to, the empirical approximation,

$$q_\gamma(\mathbf{u}) = |\tilde{\mathcal{S}}_{\text{Pareto}}|^{-1} \sum_{n=1}^{N} z_n \, \mathbb{1}[\mathbf{u} = \mathbf{u}_n], \tag{16}$$

where $\gamma = \{\mathbf{u}_n : z_n = 1\}_{n=1}^{N}$. Though on occasions when only a few observations define the Pareto front, we find that using this representation can lead to an overly exploitative strategy.

H.2. Sequence variational distributions

In this section we summarize the main variational distribution architectures considered for A-GPS VSD and CbAS. For the sequence experiments we implemented auto-regressive variational distributions of the form,

$$q_\phi(\mathbf{x}) = \text{Categ}(x_1|\text{softmax}(\phi_1)) \prod_{m=2}^{M} q_{\phi_{1:m}}(x_m|x_{1:m-1}) \quad \text{where,} \tag{17}$$

$$q_{\phi_{1:m}}(x_m|x_{1:m-1}) = \begin{cases} \text{Categ}(x_m|\text{softmax}(\text{LSTM}(x_{m-1}, \phi_{m-1:m}))), \\ \text{Categ}(x_m|\text{softmax}(\text{DTransformer}(x_{1:m-1}, \phi_{1:m}))). \end{cases}$$

For a LSTM RNN and a decoder-only transformer with a causal mask, for the latter see (**?**)Algorithm 10 & Algorithm 14]phuong2022formal for maximum likelihood training and sampling implementation details respectively. We list the configurations of the LSTM and transformer variational distributions in Table 2. We use additive positional encoding for all of these models. When using these models for priors or initialization of variational distributions, we find that over-fitting can be an issue. To circumvent this, we use early stopping for larger training datasets, or data augmentation techniques for smaller training datasets (as in the case of the Ehrlich functions).

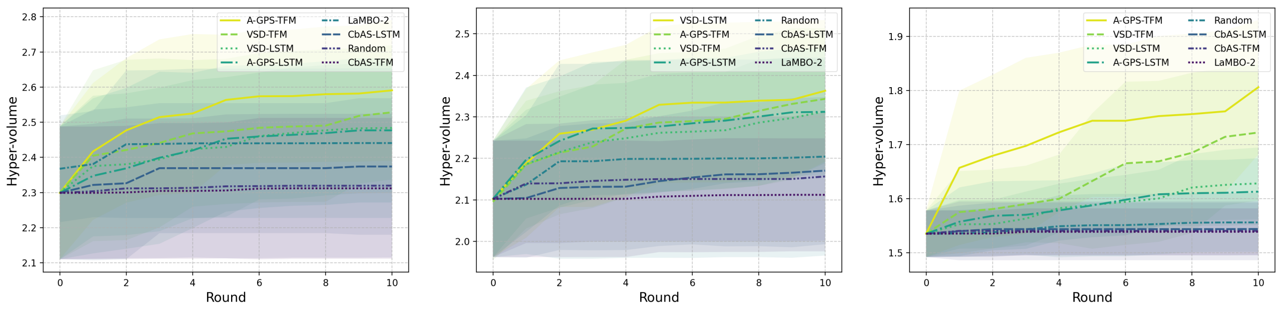
As mentioned in the text, for the conditional generative models, $q_\phi(\mathbf{x}|\mathbf{u})$, for A-GPS, we use the same architectures already discussed, but also learn a sequence prefix embedding from $\mathbf{u}$, aswell as a simple 1-hidden layer MLPs for implementing FilM (Perez et al., 2018) adaptation of the sequence token embeddings,

$$\mathbf{e}_m = \mathbf{e}_{m-1} \circ (1 + f_\alpha(\mathbf{u})) + f_\beta(\mathbf{u}), \quad \text{where} \quad \mathbf{e}_0 = f_{\text{prefix}}(\mathbf{u}). \tag{18}$$

Here $\circ$ indicates element-wise product, and f_α and f_β are the FiLM MLPs , and f_{prefix} is the prefix embedding MLP (often we find just a linear projection is adequate), and $\mathbf{e}_m$ are the LSTM or transformer embeddings. We initialize the LSTM/transformer weights in these models from their non-conditional counterparts when they are used as priors.

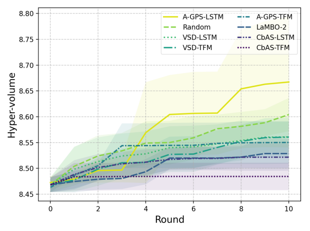
H.3. Class probability estimator architectures

For all of our experiments we share the same architecture for both $\pi_\theta(\mathbf{x}, \mathbf{u})$ and $\pi_\psi(\mathbf{x}, \mathbf{u})$. On the continuous synthetic test functions we use the MLP in Figure 7 (a), where we simply concatenate the inputs $\mathbf{x}$ and $\mathbf{u}$. Here `Skip` is a skip connection, implementing a residual layer.

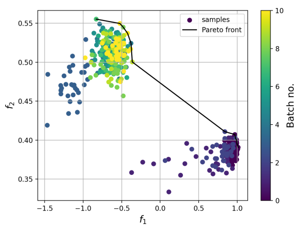


(a) $M = 15$ (b) $M = 32$ (c) $M = 64$

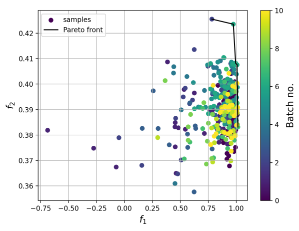
Figure 5: Ehrlich function vs. ProtGPT2 naturalness score. A-GPS, VSD and CbAS with different variational distributions; LSTM and transformer (TFM), compared against a random mutation and LaMBO-2 baselines.



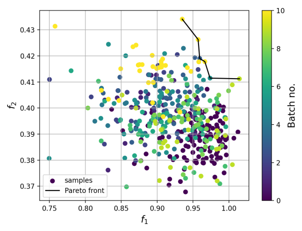
(a) Hyper-volume



(b) A-GPS-LSTM



(c) VSD-Transformer



(d) Random Mutations

Figure 6: RaSP vs. ProtGPT2 naturalness score. (b) to (d) visualize the sampled $\mathbf{y}_n$ as well as the estimated $\tilde{\mathcal{F}}_{\text{Pareto}}$. Samples are colored by batch round number, t.

Table 2: LSTM and transformer network configuration.

		Ehrlich vs. Nat			**RaSP vs. Nat.**
	$\downarrow$ **Property** / M $\rightarrow$	15	32	64	217
LSTM	Layers	3	3	3	3
	Network size	32	32	64	64
	Embedding size	10	10	10	10
	FiLM hidden size	80	80	80	80
Transformer	Layers	2	2	2	2
	Network Size	32	64	128	128
	Attention heads	1	2	3	3
	Embedding size	10	20	30	30
	FiLM hidden size	80	160	240	240

For the sequence experiments we use the convolutional architecture given in Figure 7 (b). For VSD and CbAS we simply add on another LeakyReLU and then an output linear layer. For A-GPS we concatenate $\mathbf{u}$ to output of this CNN, and then pass this concatenation into the MLP in Figure 7 (c).

I. Experimental Details

I.1. Synthetic test functions

We use `BoTorch` (Balandat et al., 2020) for the implementations of all the synthetic test functions and the baseline MOBO methods. All experiments were initialized using Latin hyper-cube sampling. The original design space is $\mathcal{X} = [0, 1]^D$ for all problems, and so we used a sigmoid transform on the designs so that $\mathcal{X} \in \mathbb{R}^D$.

For A-GPS we use the mixture model in Equation 15 for the preference direction distribution, and for the conditional generative model we use a simple MLP,

$$q_\phi(\mathbf{x}|\mathbf{u}) = \mathcal{N}\big(\mathbf{x}\big|\boldsymbol{\mu}(\mathbf{u}), \boldsymbol{\sigma}^2(\mathbf{u})\big) \,. \tag{19}$$

Here $\boldsymbol{\mu}(\mathbf{u}), \boldsymbol{\sigma}^2(\mathbf{u})$ are MLPs with 2 hidden layers of size of 32, and with skip-connections and batch normalization, making them residual networks. The rest of the settings for the experiments are given in Table 3.

Table 3: Synthetic test functions experimental settings.

Setting	**Branin-Currin**	**DTLZ2**	**ZDT3**
$N_{t=0}$	64	64	64
T	10	10	10
Replicates	5	5	5
B	5	5	5
S	128	128	128
$\mathbf{y}$ scaling	[0.005, 0.25]	None	None
$\mathbf{r}$	[0.3, 0.3]	[-1.1, -1.1]	[-0.1, -0.1]
Base BoTorch Model	`SingleTaskGP`	`SingleTaskGP`	`SingleTaskGP`

```
Sequential(
    Linear(
        in_features=D + L,
        out_features=16
    ),
    BatchNorm1D(),
    Dropout(p=0.1),
    LeakyReLU(),
    *[Skip(
        Linear(
            in_features=16,
            out_features=16
        ),
        BatchNorm1D(),
        LeakyReLU(),
    ) for _ in range(hidden_layers)],
    Linear(
        in_features=16,
        out_features=1
    ),
)
```

(a) Continuous MLP architecture

```
Sequential(
    Linear(
        in_features=128 + L,
        out_features=128
    ),
    LeakyReLU(),
    Linear(
        in_features=128,
        out_features=1
    ),
```

(c) Sequence-preference concatenation MLP architecture

```
Sequential(
    Embedding(
        num_embeddings=A,
        embedding_dim=10
    ),
    Dropout(p=0.2),
    Conv1d(
        in_channels=10,
        out_channels=16,
        kernel_size=3 or 7,
    ),
    LeakyReLU(),
    MaxPool1d(
        kernel_size=2,
        stride=2,
    ),
    Conv1d(
        in_channels=16,
        out_channels=16,
        kernel_size=3 or 7,
    ),
    LeakyReLU(),
    MaxPool1d(
        kernel_size=2,
        stride=2,
    ),
    Flatten(),
    LazyLinear(
        out_features=128
    ),
)
```

(b) Sequence CNN architecture

Figure 7: CPE architectures used for the experiments in PyTorch syntax. $\texttt{A} = |\mathcal{V}|$, $\texttt{L} = L$ corresponding to $\mathbf{y} \in \mathbb{R}^L$ and $\texttt{D} = D$ corresponding to $\mathbf{x} \in \mathbb{R}^D$ for the continuous experiments. The Ehrlich-15 functions use a kernel size of 3, all other BBO experiments use a kernel size of 7. LaMBO-2 uses the same kernel size as our CNNs for the Ehrlich functions and RaSP vs. naturalness score experiments.

I.2. Ehrlich vs. Naturalness

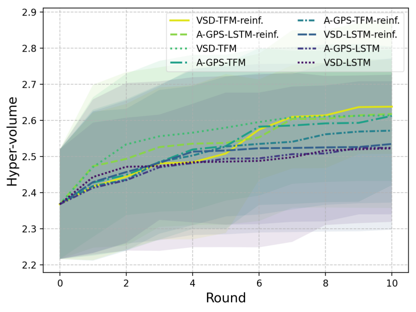
The Ehrlich vs. naturalness score benchmark was implemented using the `poli` benchmarking library (González-Duque et al., 2024), where we implemented our own ProtGPT2-based naturalness black box, and used the inbuilt Ehrlich function (Stanton et al., 2024) (not the holo version). For A-GPS we use the aforementioned generative and discriminative models, otherwise the settings are given in Table 4. We use a modified version of the LaMBO-2 algorithm (Gruver et al., 2023) from `poli-baselines` (González-Duque et al., 2024).

I.3. RaSP vs. Naturalness

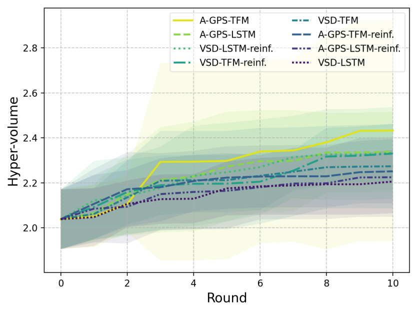
Similarly to the Ehrlich experiment, we use `poli` (González-Duque et al., 2024) to implement this experiment, were we use the inbuilt RaSP black-box (Blaabjerg et al., 2023) with additive mutation effects. All designs were based off the DsRED.M1 protein in (Stanton et al., 2022), and the initial training data consisted of 128 randomly mutated versions of this sequence (max 3 mutations), scored by the RaSP predictor. The generative and discriminative models are discussed in the preceding sections, and extra experimental configuration is in Table 4.

Table 4: Erhlich and RaSP vs. naturalness score experimental settings.

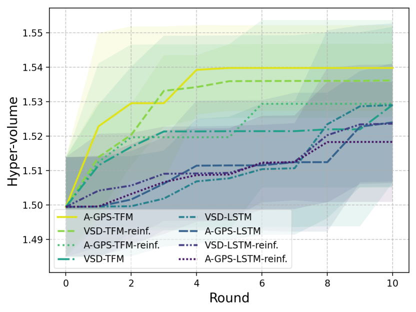
↓ **Setting** / M →	**Ehrlich vs. Nat.** 15	32	64	**RaSP vs. Nat.** 217
$N_{t=0}$	128	128	128	128
T	10	10	10	10
Replicates	5	5	5	5
B	32	32	32	32
S	256	256	256	256
$\mathbf{r}$	[-1, -1]	[-1, -1]	[-1, -1]	[-5, -1]



(a) $M = 15$



(b) $M = 32$



(c) $M = 64$

Figure 8: Ehrlich function vs. ProtGPT2 naturalness score ablation. On-policy gradient estimators ('-reinf.') vs. the importance weighted off-policy estimators for the A-GPS and VSD methods.

I.4. Computational resources

The synthetic function experiments were run on an Apple Macbook Pro (M2) with 32GB of RAM. All sequence experiments were run on a cluster with NVIDIA H100 GPUs. All of our models could easily fit on one GPU, and typically took less than 1 hour to complete the experiment. See the ablation study in Sec. J.1 for more detailed timing information.

J. Ablation Studies

In this section we test some of the architectural decisions we have made when designing A-GPS.

J.1. On-policy vs. off-policy gradients

We introduce a new gradient estimator in Equation 11 based on off-policy importance weighting approximations to the on-policy gradient estimator used by (Steinberg et al., 2025). To test its efficacy, we re-run the Ehrlich vs. naturalness score experiments with this new estimator and the original on-policy variant. We report performance in Figure 8 and runtimes in Table 5. There does not seem to be a consistent difference between the two gradient estimators in terms of hyper-volume performance, but runtime is significantly lower for the off-policy estimator, being almost an order of magnitude less for the longer, $M = 64$, length sequences.

Note that in Figure 8c we limited the maximum naturalness score to 0.5, to prevent the algorithms from simply maximising only naturalness over Ehrlich score for this sequence length (unlike the original experiment).

J.2. Empirical vs. parameterized preference direction distribution

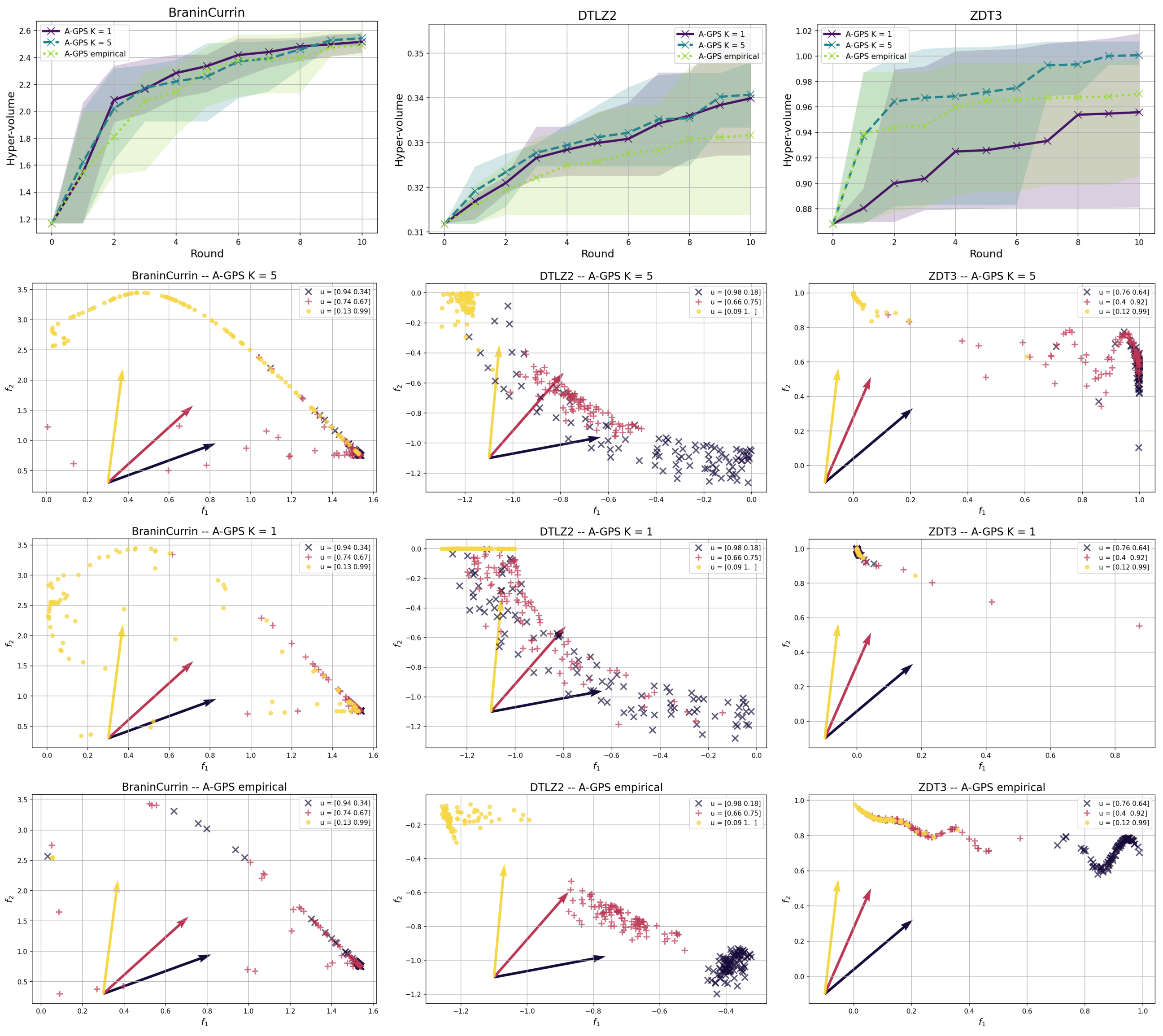
For all of our experiments we use the constrained mixture of Normal distributions ($K = 5$), Equation 15, as our parameterized preference directions distribution, $q_\gamma(\mathbf{u})$. We now wish to validate this choice by comparing it to two simpler alternatives: a single constrained Normal distribution ($K = 1$), and the empirical distribution in Equation 16.

We make these comparisons on the synthetic test functions, as their Pareto fronts are the cleanest to visualize. As can be seen in the results in Figure 9, there is not a large performance difference for the empirical or mixture parameterizations – both seem to be valid choices for these experiments. Though the empirical parameterization may perform slightly more

Table 5: Ehrlich function vs. ProtGPT2 naturalness score ablation. Run time comparison for the on-policy gradient estimators ('-reinf.') vs. the importance weighted off-policy estimators for the A-GPS and VSD methods. All times are in minutes.

M	**15**			**32**			**64**		
Method	mean	min	max	mean	min	max	mean	min	max
A-GPS-LSTM	4.78	4.63	5.14	5.52	4.90	6.29	6.66	5.70	8.54
A-GPS-LSTM-reinf.	11.73	11.35	11.92	18.61	18.44	18.79	32.10	29.98	32.77
A-GPS-TFM	7.79	7.61	8.04	10.38	8.51	11.06	9.52	7.51	11.49
A-GPS-TFM-reinf.	23.44	23.16	23.72	40.02	39.78	40.59	76.93	73.23	82.25
VSD-LSTM	3.96	3.91	4.07	4.62	4.39	4.77	6.11	4.63	7.62
VSD-LSTM-reinf.	10.21	10.15	10.25	16.59	16.37	16.77	27.06	26.38	27.56
VSD-TFM	6.96	6.81	7.25	8.02	7.17	8.57	7.97	7.11	8.27
VSD-TFM-reinf.	22.81	22.66	23.00	39.34	38.86	40.05	70.86	69.85	71.68

weakly on the DTLZ2 and ZDT3 functions. However, we do see a potential decrease in performance with $K = 1$ – and the sampled sequences seem to have worse alignment using this representation compared to the mixture and empirical options.



(a) Negative Branin-Currin (b) DTLZ2 (c) ZDT3

Figure 9: Ablation experimental results on three test functions commonly used in the MOBO literature. The top row reports HVI per round, three bottom row demonstrates amortized preference conditioning using the different preference direction distribution parameterizations.