Optimistic Games for Combinatorial Bayesian Optimization with Applications to Protein Design

Melis İlayda Bal MBAL@MPI-SWS.ORG
MPI-SWS

Pier Giuseppe Sessa PIERGIUSEPPE.SESSA@INF.ETHZ.CH

Mojmír Mutný MMUTNY@INF.ETHZ.CH

Andreas Krause KRAUSEA@ETHZ.CH
ETH Zürich

Abstract

Bayesian optimization (BO) is a powerful framework to optimize black box expensive-to-evaluate functions via sequential interactions. In several important problems (e.g. drug discovery, circuit design, neural architecture search, etc.), though, such functions are defined over *combinatorial and unstructured* spaces. This makes existing BO algorithms not feasible due to the intractable maximization of the acquisition function to find informative evaluation points. To address this issue, we propose GameOpt, a novel game-theoretical approach to combinatorial BO. GameOpt establishes a cooperative game between the different optimization variables and computes informative points to be game *equilibria* of the acquisition function. These are stable configurations from which no variable has an incentive to deviate – analog to local optima in continuous domains. Crucially, this allows us to efficiently break down the complexity of the combinatorial domain into individual decision sets, making GameOpt scalable to large combinatorial spaces. We demonstrate the application of GameOpt to the challenging *protein design* problem and validate its performance on two real-world protein datasets. Each protein can take up to 20^X possible configurations, where X is the length of a protein, making standard BO methods unusable. Instead, our approach iteratively selects informative protein configurations and very quickly discovers highly active protein variants compared to other baselines.

Keywords: Combinatorial BO, Game Theory, Gaussian Processes, Protein Design

1. Introduction

Many scientific and engineering problems such as drug discovery (Negoescu et al., 2011), neural architecture search (Kandasamy et al., 2018), or circuit design (Lyu et al., 2018) involve the optimization of expensive-to-evaluate black-box functions over combinatorial unstructured spaces involving binary, integer-valued, and categorical variables. As a concrete example, consider the *protein design* problem *i.e.* finding the optimal amino acid sequence to maximize the functional capacity (fitness) of the protein. Such fitness function can be elucidated only from queries involving real-world protein synthesis experiments. Moreover, exhaustive exploration is infeasible for both traditional lab methods and computational techniques (Romero et al., 2013) due to *combinatorial explosion*: a typical protein has 300 amino acid sites, each to be filled with one of twenty natural amino acids, yielding 20^{300} candidate variants.

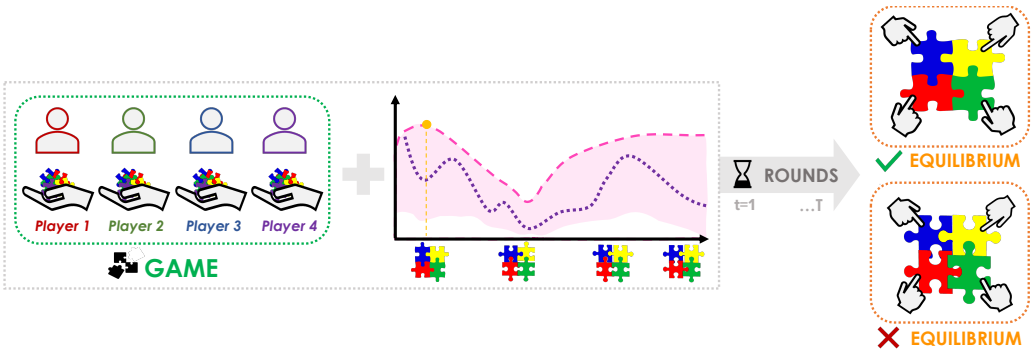


Figure 1: Illustration of GameOpt. GameOpt defines a game among the decision variables, where game rewards are represented by the upper confidence bound (UCB) function. This decouples the combinatorial decision space into individual decision sets and allows GameOpt to efficiently compute game equilibria, *i.e.* analog of local maxima of the acquisition function.

A standard framework for optimizing black-box functions with minimal evaluations is Bayesian optimization (BO) (Mockus, 1974), which has proven successful in a variety of domains. BO constructs a probabilistic surrogate model as a representation of the underlying black-box function *e.g.* using Gaussian Processes (GPs) (Rasmussen et al., 2006). Then, it iteratively selects informative decision points which are typically the maximizers of a designed acquisition function. When considering combinatorial domains, however, standard BO methods are intractable since the latter step requires exhaustive search over the whole combinatorial space (*e.g.* of size 20^{300} in the context of proteins) without further assumptions.

To address this challenge, in this work, we propose GameOpt, a novel game-theoretical framework for combinatorial BO. To circumvent the intractable maximization of an acquisition function, GameOpt defines a cooperative game between the discrete domain variables and, at each round, selects informative points to be game *equilibria* of the acquisition function (in this work we take this to be the Upper Confidence Bound (UCB) function). These are stable configurations from which no agent (variable) has an incentive to deviate and can be thought of as local optima of the underlying problem. For an overview of the method, see Figure 1.

Contributions We make the following contributions:

- We propose GameOpt, a **novel** game-theoretical BO framework for large combinatorial and unstructured search spaces. GameOpt computes informative evaluation points as the equilibria (*i.e.* local optima) of a cooperative game between the discrete variables. This overcomes the scalability issues of maximizing acquisition functions over combinatorial domains. GameOpt is a **flexible** procedure where the resulting per-iteration game can be solved by any readily available game strategy or solver.
- We show the applicability of GameOpt to the challenging **protein design** problem, involving search spaces of categorical inputs. There, GameOpt advances the protein design process by mimicking natural evolution via a game between protein sites.
- We **experimentally** validate the performance of GameOpt on two real-world protein design problems based on human binding protein GB1 (Wu et al., 2016; Olson et al., 2014). We show that GameOpt converges faster *i.e.* it requires less number of BO

iterations to identify highly binding protein variants compared to baseline methods such as classical directed evolution.

2. Problem Statement and Background

Problem Statement We consider the problem of optimizing a costly-to-evaluate, black-box function $f : \mathcal{X} \to \mathbb{R}$ over a combinatorial unstructured space $\mathcal{X}$. Without loss of generality, let each element $x \in \mathcal{X}$ be represented by n discrete variables $x^1, x^2, \ldots, x^n$, where each x^i takes values from a set $\mathcal{X}^{(i)}$, this makes the domain of $n \geq 1$ variables $\mathcal{X} = \mathcal{X}^{(1)} \times \ldots \mathcal{X}^{(n)}$. Assuming $|\mathcal{X}^{(i)}| = k$, $\forall i$, the size of the combinatorial space $\mathcal{X}$ is k^n.

As a concrete motivating example, consider the protein design problem considered in Sec. 4. There, $f(x)$ corresponds to the fitness value of the designed amino acid sequence x, and each x can take 20^n values where n is the number of protein sites. Moreover, a (noisy) evaluation $f(x)$ is a labor-intensive process, requiring extensive efforts and specialized laboratory equipment.

Bayesian Optimization (BO) and Gaussian Processes (GPs) Bayesian Optimization (Mockus, 1974) is a powerful framework for optimizing complex, noisy, and expensive-to-evaluate functions. BO leverages Bayesian inference to model the underlying function with a surrogate *e.g.* a Gaussian Process (GP) and iteratively selects evaluation points that are the most informative in terms of reducing uncertainty or enhancing model performance.

Formally, a Gaussian Process $\mathcal{GP}(\mu(\cdot), k(\cdot,\cdot))$ over domain $\mathcal{X}$ is specified by a prior mean function $\mu(x) : \mathcal{X} \to \mathbb{R}$ and a covariance function $k(x, x') : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, denoted by $f(x) \sim \mathcal{GP}(\mu(x), k(x, x'))$, where $f(x)$ represents the function value at input x. The mean function $\mu(x)$ characterizes the expected value of $f(x)$ *i.e.* $\mathbb{E}\left[f\left(x_i\right)\right] = \mu\left(x_i\right), \forall x_i \in \mathcal{X}$, and the kernel (covariance function) $k(x, x') = \mathbb{E}\left[\left(f\left(x\right) - \mu\left(x\right)\right)\left(f\left(x'\right) - \mu\left(x'\right)\right)\right]$ governs the correlation between $f(x)$ and $f(x')$ for any pair of inputs $x, x' \in \mathcal{X}$. Given a set of observed data points X and their corresponding vector of noisy observations $Y = f(X) + \epsilon$ with Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$, and a GP prior defined by $\mathcal{GP}(\mu(x), k(x, x'))$, the posterior distribution of the GP given new observations $X_\dagger$ is again Gaussian $p(f_\dagger \mid X, X_\dagger, f) = \mathcal{N}(\mu_\dagger, \sigma^2_\dagger)$ with posterior mean $\mu_\dagger = K(X_\dagger, X)[K(X, X) + \sigma^2 I]^{-1} Y$ and posterior variance $\sigma^2_\dagger = K(X_\dagger, X_\dagger) - K(X_\dagger, X)[K(X, X) + \sigma^2 I]^{-1} K(X, X_\dagger)$, where K and I are the kernel and identity matrices, respectively.

To maximize f, BO algorithms iteratively select evaluation points so as to balance exploration and exploitation. Typically, at each iteration, they select the maximizer of a given *acquisition function* such as the widely-adopted Upper-confidence bound (UCB) (Srinivas et al., 2009) function. Given a $\mathcal{GP}$ model, the UCB function is defined as

$$\text{UCB}(\mathcal{GP}, x) = \mu(x) + \beta\sigma(x), \tag{1}$$

where $\mu(x)$ and $\sigma(x)$ are the posterior mean and standard deviation at point x according to $\mathcal{GP}$, and $\beta \in \mathbb{R}$ is tunable width.

While standard BO methods can efficiently optimize $\text{UCB}(\mathcal{GP}, \cdot)$ in relatively-sized finite or continuous domains, they become very soon intractable in the case of combinatorial unstructured domains, such as the space of possible amino acid sequences. In the next section, we propose GameOpt, a novel BO approach that circumvents such prohibitive difficulty.

Algorithm 1 GameOpt

Input: GP prior $\mathcal{GP}^0(\mu_0, k(\cdot,\cdot))$, initial data $\mathcal{D}_0 = \{(x_i, y_i = f(x_i) + \epsilon)\}$, batch size $B > 0$.

for iteration $k = 1, 2, \ldots, K$ **do**
  Construct game with reward function $\text{UCB}(\mathcal{GP}^{k-1}, \cdot) : \prod_{i=1}^{n} \mathcal{X}^{(i)} \to \mathbb{R}$
  Compute batch of B equilibria $\{x_{k,i}\}_{i=1}^{B}$ of the above. ▷ Equilibrium subroutine
  Obtain evaluations $y_{k,i} = f(x_{k,i}) + \epsilon_{k,i}, \quad \forall i = 1, \ldots, B$
  Update $\mathcal{D}_k \leftarrow \mathcal{D}_{k-1} \cup \{(x_{k,i}, y_{k,i})\}_{i=1}^{B}$
  Posterior update of model $\mathcal{GP}^k$ with $\mathcal{D}_k$
end for
return $x_K^\star \leftarrow \arg\max_{(x,y)\in\mathcal{D}_K} y$ ▷ Best-so-far

3. The GameOpt approach

In a nutshell, the proposed GameOpt (Optimistic Games) approach circumvents the combinatorial optimization of the UCB function by defining a *cooperative game* among the n input variables and computes the associated equilibria as candidate evaluation points. More formally, at each iteration k, GameOpt defines a cooperative game (Fudenberg and Tirole, 1991) involving $\mathcal{N} = \{1, 2, \ldots, n\}$ players, each player i taking actions in the discrete set $\mathcal{X}_i$. In such a game, the players' interests are aligned towards the goal of maximizing the UCB function $\text{UCB}(\mathcal{GP}^k, \cdot) : \prod_{i=1}^{N} \mathcal{X}^{(i)} \to \mathbb{R}$, where $\mathcal{GP}^k$ is the current GP estimate at iteration k. Thus, it can be interpreted as an optimistic game w.r.t. to the true unknown f. In such a game, the goal of the players is to compute game *equilibria*, defined as follows.

Def 1 (Equilibrium)**.** Let $v : \mathcal{X} \to \mathbb{R}$ be the game reward function. A joint strategy profile $x_{\text{eq}} = (x_{\text{eq}}^1, \ldots, x_{\text{eq}}^n)$ is an equilibrium if, for every player $i \in \mathcal{N}$, $v(x_{\text{eq}}^i, x_{\text{eq}}^{-i}) \geq v(x^i, x_{\text{eq}}^{-i}), \forall x_i \in \mathcal{X}_i$, where x_{eq}^{-i} is the joint equilibrium strategy of all players except i.

The existence of such equilibrium point(s) is guaranteed for finite games with finite sets of players, actions, and payoffs (Fudenberg and Tirole, 1991). Moreover, efficient polynomial-time equilibrium-finding methods can be employed, such as Iterative Best-Response (IBR), where players update their actions sequentially, or simultaneous multiplicative weights updates such as the Hedge (Freund and Schapire, 1997) algorithm. We report these two possible strategies in Algorithms 2 and 3 in Appendix A. Intuitively, they achieve this by breaking down the complex decision space into individual decision sets, as illustrated in Figure 1. Our overall approach is summarized in Algorithm 1, where we allow to compute a batch of $B > 1$ equilibria. Such a batch is evaluated by f, the GP model is updated accordingly, and a new game is defined at the next iteration based on the updated posterior.

Local Optimality Within GameOpt, each player strategically selects actions to maximize their collective payoff, much like seeking local optima in a continuous multi-dimensional function (see Figure 1). In continuous optimization, a local optimum is a point, where there is no direction that leads to an improvement, similarly, as in our framework there is not a player that can unilaterally improve the value of the collective pay-off. In essence, seeking equilibria is an analog of seeking local optima of a continuous acquisition function, and our game-based approach allows us to effectively pinpoint them within a combinatorial space.

3.1 Related work

While there exist rather few works in the area, existing combinatorial BO methods either target surrogate modeling with discrete variables (Baptista and Poloczek, 2018; Oh et al., 2019; Garrido-Merchán and Hernández-Lobato, 2020; Kim et al., 2021) or optimizing acquisition function within discrete spaces (Baptista and Poloczek, 2018; Deshwal et al., 2020, 2021a,b; Khan et al., 2023). However, they often require a parametric surrogate model with higher-order interaction specifications for combinatorial structures (Baptista and Poloczek, 2018) or domain-specific knowledge (Deshwal et al., 2020). In contrast, GameOpt relies on a non-parametric surrogate model, without the need for domain-specific knowledge.

Closest to ours is (Daulton et al., 2022), which also targets optimizing the acquisition function in high-cardinality discrete/mixed search spaces via a probabilistic reparameterization (PR) that maximizes the expectation of the acquisition function. However, PR fails at being tractable since it requires evaluating the expectation over the joint distribution of all decision variables, requiring combinatorially many elements to be summed. An accurate estimate would need a similar number of samples proportional to a combinatorially large number without a special structure. In contrast, GameOpt treats each variable *independently* (potentially in parallel) within the game, keeping the values of the remaining variables fixed during each strategy update. We use PR as a baseline to evaluate our approach in Sec. 4.

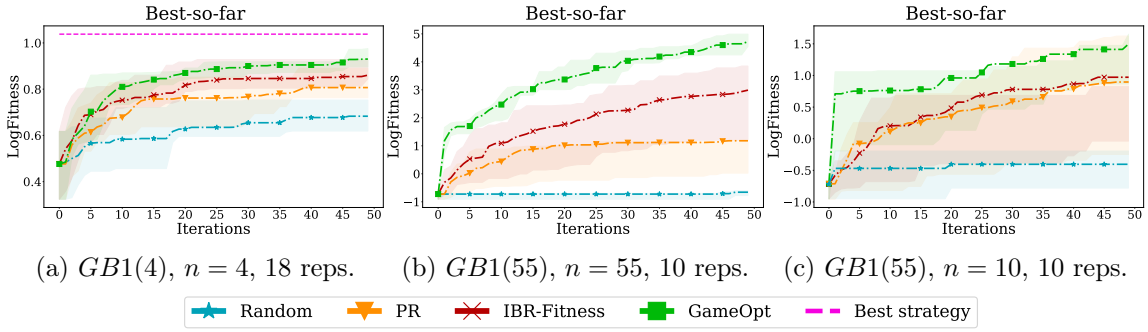
Recently, the interplay between BO and game theory has been explored by the line of works (Sessa et al., 2019, 2022), but its connection with combinatorial BO is novel.

4. Application to Protein Design

In this section, we specialize the GameOpt framework to protein design. In this context, computing game equilibria follows the natural principle of beneficial mutants and mirrors the proteins' mutation and selection process. Inspired by this, in Algorithm 4 (Appendix B) we provide a tailored version of GameOpt for protein design which computes equilibria via an iterated best response rule. We showcase its performance in two real-world datasets.

In protein design context, GameOpt establishes a cooperative game among the different protein sites $i \in \{1, \dots, n\}$, where n is the length of the protein sequence. Each site i chooses an amino acid from the set $\mathcal{X}_i = \{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\}$, representing the concept of *mutations*. Their objective is to converge to a highly rewarding protein sequence, as measured by the GP-predicted UCB score. This mirrors the *selection* phase in *evolutionary search*, providing a *directed* approach to optimization. Compared to classical evolutionary methods (see Appendix B for an overview), though, GameOpt mimics evolution at each interaction using the surrogate UCB function.

Datasets We empirically evaluate GameOpt on a real-world protein design problem: protein G domain B1, $GB1$, binding affinity to an antibody IgG-FC (KA) on two datasets $GB1(4)$ (Wu et al., 2016) and $GB1(55)$ (Olson et al., 2014), with sequence length 4 and 55, respectively. The former is fully combinatorial *i.e.* covering fitness measurements of 20^4 variants. Here, each protein site is treated as a player in the GameOpt. The latter is non-exhaustive, including only 2-point mutations of $GB1$. Thus, an MLP having $R^2 = 0.93$ on a test set is trained and treated as the ground truth fitness for the fully combinatorial



(a) $GB1(4)$, $n = 4$, 18 reps. (b) $GB1(55)$, $n = 55$, 10 reps. (c) $GB1(55)$, $n = 10$, 10 reps.

Figure 2: Convergence speed of methods in terms of log fitness value of the best-so-far protein throughout BO iterations, under batch size $B = 5$. Each point is the average of multiple replications initiated with different training sets having 100 and 1000 protein variants, for $GB1(4)$ and $GB1(55)$. Similarly, error bars are interquartile ranges averaged over replications. In all experiments GameOpt discovers better protein sequences at a much faster rate.

dataset. For $GB1(55)$, we also consider a modified setup where "only" 10 sites can be mutated. Further experimental details are in Appendix D.

Baselines We benchmark GameOpt against the following baselines: (1) IBR-Fitness, which mimics directed evolution (Arnold, 1998) through a series of local searches on the fitness landscape, iteratively selecting the B best-responses based on log fitness criterion; (2) PR (Daulton et al., 2022), a state-of-the-art discrete/mixed BO approach picking B points using the expected UCB criterion, and (3) Random baseline randomly sampling B random sequences at each iteration. Further method details are in Appendix C.

We assess our method using two key metrics: convergence speed and sampled batch diversity (*i.e.* the degree of distinctiveness among newly acquired samples in comparison to the original data point particularly in the context of the input space) for BO evaluation. Convergence speed is tracked by the log fitness value of the best-so-far discovered protein variant across BO iterations. We provide the evaluation in terms of sampled batch diversity in Appendix E.

Results GameOpt consistently outperforms baselines in all experiments, discovering higher log fitness protein sequences faster as shown in Figure 2. While initially slightly surpassed by IBR-Fitness in $GB1(4)$ setting, GameOpt can more efficiently explore and samples diverse points (see Figure 3 in Appendix E), discovering high fitness proteins faster. Notably, while IBR-Fitness performs best-responses on the true log fitness function, GameOpt simulates best-response dynamics directly on the UCB model, allowing to compute equilibria at each iteration. In $GB1(55)$, GameOpt excels in identifying high log fitness protein sequences even from the start.

In addition to being outperformed in all experiments, PR also comes with higher computational demands. As highlighted in Sec. 3.1, PR relies on the expected UCB as the acquisition function, requiring expectation computation across players set and amino acid choices. This makes its performance contingent on accurately estimating expected UCB through combinatorially many sequence samples. In contrast, GameOpt efficiently finds stable outcomes by breaking down the combinatorial search space into individual decision sets, resulting in a more manageable process. Further discussion on the performance of methods can be found in Appendix E.

References

Frances H Arnold. Design by directed evolution. *Accounts of chemical research*, 31(3): 125–131, 1998.

Ricardo Baptista and Matthias Poloczek. Bayesian optimization of combinatorial structures. In *International Conference on Machine Learning*, pages 462–471. PMLR, 2018.

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. 01 2006.

Lixue Cheng, Ziyi Yang, Changyu Hsieh, Benben Liao, and Shengyu Zhang. Odbo: Bayesian optimization with search space prescreening for directed protein evolution. *arXiv preprint arXiv:2205.09548*, 2022.

Samuel Daulton, Xingchen Wan, David Eriksson, Maximilian Balandat, Michael A Osborne, and Eytan Bakshy. Bayesian optimization over discrete and mixed spaces via probabilistic reparameterization. *Advances in Neural Information Processing Systems*, 35:12760–12774, 2022.

Aryan Deshwal, Syrine Belakaria, Janardhan Doppa, and Alan Fern. Optimizing discrete spaces via expensive evaluations: A learning to search framework. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:3773–3780, 04 2020.

Aryan Deshwal, Syrine Belakaria, and Janardhan Rao Doppa. Bayesian optimization over hybrid spaces. In *International Conference on Machine Learning*, pages 2632–2643. PMLR, 2021a.

Aryan Deshwal, Syrine Belakaria, and Janardhan Rao Doppa. Mercer features for efficient combinatorial bayesian optimization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):7210–7218, May 2021b.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Drew Fudenberg and Jean Tirole. *Game theory*. MIT press, 1991.

Eduardo C Garrido-Merchán and Daniel Hernández-Lobato. Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes. *Neurocomputing*, 380:20–35, 2020.

Nikolaus Hansen. The cma evolution strategy: a comparing review. *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms*, pages 75–102, 2006.

Kirthevasan Kandasamy, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric P Xing. Neural architecture search with bayesian optimisation and optimal transport. *Advances in Neural Information Processing Systems*, 31, 2018.

Asif Khan, Alexander I Cowen-Rivers, Antoine Grosnit, Philippe A Robert, Victor Greiff, Eva Smorodina, Puneet Rawat, Rahmad Akbar, Kamil Dreczkowski, Rasul Tutunov, et al. Toward real-world automated antibody design with combinatorial bayesian optimization. *Cell Reports Methods*, 3(1), 2023.

Jungtaek Kim, Michael McCourt, Tackgeun You, Saehoon Kim, and Seungjin Choi. Bayesian optimization with approximate set kernels. *Machine Learning*, 110:857–879, 2021.

Andre KY Low, Flore Mekki-Berrada, Aleksandr Ostudin, Jiaxun Xie, Eleonore Vissol-Gaudin, Yee-Fun Lim, Abhishek Gupta, Qianxiao Li, Yew Soon Ong, Saif A Khan, et al. Evolution-guided bayesian optimization for constrained multi-objective optimization in self-driving labs. *ChemRxiv*, 2023.

Wenlong Lyu, Fan Yang, Changhao Yan, Dian Zhou, and Xuan Zeng. Batch bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design. In *International Conference on Machine Learning*, pages 3306–3314. PMLR, 2018.

Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alex Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in Neural Information Processing Systems*, 34:29287–29303, 2021.

Jonas Mockus. On bayesian methods for seeking the extremum. In *Optimization Techniques*, 1974.

Diana M Negoescu, Peter I Frazier, and Warren B Powell. The knowledge-gradient algorithm for sequencing experiments in drug discovery. *INFORMS Journal on Computing*, 23(3): 346–363, 2011.

Changyong Oh, Jakub Tomczak, Efstratios Gavves, and Max Welling. Combinatorial bayesian optimization using the graph cartesian product. *Advances in Neural Information Processing Systems*, 32, 2019.

C Anders Olson, Nicholas C Wu, and Ren Sun. A comprehensive biophysical description of pairwise epistasis throughout an entire protein domain. *Current biology*, 24(22):2643–2651, 2014.

Patrick C Phillips. Epistasis—the essential role of gene interactions in the structure and evolution of genetic systems. *Nature Reviews Genetics*, 9(11):855–867, 2008.

Carl Edward Rasmussen, Christopher KI Williams, et al. *Gaussian processes for machine learning*, volume 1. Springer, 2006.

Philip A. Romero and Frances H. Arnold. Exploring protein fitness landscapes by directed evolution. *Nature Reviews Molecular Cell Biology*, 10:866–876, 2009.

Philip A Romero, Andreas Krause, and Frances H Arnold. Navigating the protein fitness landscape with gaussian processes. *Proceedings of the National Academy of Sciences*, 110 (3):E193–E201, 2013.

Pier Giuseppe Sessa, Ilija Bogunovic, Maryam Kamgarpour, and Andreas Krause. No-regret learning in unknown games with correlated payoffs. *Advances in Neural Information Processing Systems*, 32, 2019.

Pier Giuseppe Sessa, Maryam Kamgarpour, and Andreas Krause. Efficient model-based multi-agent reinforcement learning via optimistic equilibrium computation. In *International Conference on Machine Learning*, pages 19580–19597. PMLR, 2022.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Bruce J Wittmann, Kadina E Johnston, Zachary Wu, and Frances H Arnold. Advances in machine learning for directed evolution. *Current opinion in structural biology*, 69:11–18, 2021.

Nicholas C. Wu, Lei Dai, Anders Olson, James O. Lloyd-Smith, and Ren Sun. Adaptation in protein fitness landscapes is facilitated by indirect paths. *eLife*, 5, 2016.

Zachary Wu, S. B. Jennifer Kan, Russell D. Lewis, Bruce J. Wittmann, and Frances H. Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences*, 116(18):8852–8858, 2019.

Kevin K Yang, Zachary Wu, and Frances H Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694, 2019.

Appendix A. Equilibrium finding subroutines

With IBR, evaluation points are generated from the best responses (BRs) as provided in Algorithm 2. Concretely, under the cooperative game setting outlined in Sec. 3 and given $\mathcal{GP}$-predicted UCB function, each player simultaneously best responds to the joint strategy from the previous round. Subsequently, the strategy that maximizes the predicted UCB is executed for that particular round.

Algorithm 2 ITERATIVEBESTRESPONSE (IBR)

Input: Domain $\mathcal{X}$, payoff $v : \mathcal{X} \to \mathbb{R}$, players $\mathcal{N}$, parameters B, β.

$\mathbf{x}_0^{br} \leftarrow$ random joint strategy, $\mathbf{x}_0^{br} \in \mathcal{X}$

for round $t = 1, 2, \ldots, T$ **do** ▷ BR game

  $\mathcal{X}_t^{br} \leftarrow \left\{(x^{i,br}, x_{t-1}^{-i,br}),\ \text{such that } x^{i,br} = \arg\max_{x \in \mathcal{X}^{(i)}} v(x, x_{t-1}^{-i,br})\right\}_{i=1}^{n}$

  Play $x_t^{br} \leftarrow \arg\max_{x_t \in \mathcal{X}_t^{br}} [v(x_t)]$

end for

return $\mathbf{x}_T^{br}$ ▷ Equilibrium

Using HEDGE (Freund and Schapire, 1997), we cast the sampling batch step of GAMEOPT as an instance of adversarial online learning (Cesa-Bianchi and Lugosi, 2006) with multiple learners. Here, each player selects a strategy based on their available options, without knowledge of the payoff function selected by an adversary (nature). After observing the joint payoff, players' strategies are re-weighted based on past performance. Through repeated rounds of play and re-weighting, the empirical frequency of play forms a Coarse Correlated Equilibrium (CCE) (Cesa-Bianchi and Lugosi, 2006).

Algorithm 3 HEDGE

Input: Domain $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}^{(i)}$ with $\mid \mathcal{X}^{(i)} \mid = K$, payoff $v : \mathcal{X} \to \mathbb{R}$, players $\mathcal{N}$, parameters η, β, t_{last}.

Initialize weights $\boldsymbol{w}_1 \leftarrow \frac{1}{K}[1, \ldots, 1] \in \mathbb{R}^{|\mathcal{N}| \times K}$

for round $t = 1, 2, \ldots T$ **do** ▷ Compute CCE

  Sample $x_t^i \sim \boldsymbol{w}_1^i, \forall i \in \mathcal{N}$

  Set joint strategy $x_t \leftarrow \bigcup_{i \in \mathcal{N}} x_t^i$

  for player $i \in \mathcal{N}$ **do** ▷ Players' payoff

    $\ell_{x_t^{-i}} \leftarrow [v(x_t^{j,-i})]_{\forall j \in \mathcal{X}^{(i)}}$, where $x_t^{j,-i} = j \cup \{\bigcup_{i' \in \mathcal{N} \setminus \{i\}} x_t^{i'}\}, \forall j \in \mathcal{X}^{(i)}$

    Set $\boldsymbol{w}_{t+1}^i \propto \boldsymbol{w}_t^i \exp(\eta \ell_{x_t^{-i}})$

  end for

end for

return Uniform$\{x_1, \ldots, x_T\}$ ▷ Equilibrium

Appendix B. GAMEOPT FOR PROTEIN DESIGN

The core concept of the GAMEOPT framework is inspired by the principles of natural evolution. In protein design, achieving equilibrium of a cooperative game over protein sites mirrors

the iterative mutation and selection process in evolution. Where it converges to beneficial mutant sequences, can be thought of as equilibrium of the game. Given that protein search spaces align well with the domain GameOpt works on, we introduce a specialized version of GameOpt, tailored for protein design applications.

Algorithm 4 GameOpt for Protein Design

Input: GP prior $\mathcal{GP}^0(\mu_0, k(\cdot,\cdot))$, initial data $\mathcal{D}_0 = \{(x_i, y_i = f(x_i) + \epsilon)\}$, protein sites $\mathcal{N}$, batch size $B > 0$, parameter β.

for iteration $k = 1, 2, \ldots, K$ **do**
  Construct game with reward function $\text{UCB}(\mathcal{GP}^{k-1}, \beta, \cdot) : \prod_{i=1}^{n} \mathcal{X}^{(i)} \to \mathbb{R}$
  for $b = 1, 2, \ldots, B$ **do**
    $x_0^{br} \leftarrow$ random starting protein sequence, $x_0^{br} \in \mathcal{X}$
    for round $t = 1, 2, \ldots, T$ **do** ▷ BR game
      $\mathcal{X}_t^{br} \leftarrow \left\{(x^{i,br}, x_{t-1}^{-i,br}),\ \text{such that } x^{i,br} = \arg\max_{x \in \mathcal{X}^{(i)}} \text{UCB}(x, x_{t-1}^{-i,br})\right\}_{i=1}^{n}$
      Play $x_t^{br} \leftarrow \arg\max_{\mathbf{x}_t \in \mathcal{X}_t^{br}} [\text{UCB}(x_t)]$
    end for
    Collect equilibrium protein sequence $x_{k,b} \leftarrow x_T^{br}$
  end for
  Obtain fitness evaluations $y_{k,i} = f(x_{k,i}) + \epsilon_{k,i}, \quad \forall i = 1, \ldots, B$
  Update $\mathcal{D}_k \leftarrow \mathcal{D}_{k-1} \cup \{(x_{k,i}, y_{k,i})\}_{i=1}^{B}$
  Posterior update of model $\mathcal{GP}^k$ with $\mathcal{D}_k$
end for
return $x_K^\star \leftarrow \arg\max_{(x,y) \in \mathcal{D}_K} y$ ▷ Best-so-far

Evolutionary search A considerable line of work (Arnold, 1998; Hansen, 2006; Romero and Arnold, 2009; Yang et al., 2019; Deshwal et al., 2020; Cheng et al., 2022; Low et al., 2023) centers around evolutionary search algorithms for optimizing black-box functions. Within combinatorial amino-acid sequence space, the highly regarded technique, *directed evolution* (Arnold, 1998; Romero and Arnold, 2009), draws inspiration from natural evolution and identifies local optima through a series of repeated random searches, characterized by controlled iterative cycles of mutation and selection. Expanding upon this, machine learning-guided variants (Yang et al., 2019; Wittmann et al., 2021) mitigate the sample-inefficiency and intractability concerns associated with directed evolution. In general, these methods are not data-driven in the sense they do not use the whole extent of past data. They focus on the best variant found so far or a selection of thereof and propose a random search from thereon. Our approach uses all the data to create an estimate of the fitness landscape and utilize it to simulate a cooperative evolution.

Appendix C. Baselines

As explained in Sec. 4, we empirically evaluate GameOpt with the Ibr algorithm for protein design, comparing it to several baselines: IBR-Fitness, inspired by directed evolution (Algorithm 5), Random (Algorithm 6), samples evaluation points randomly, and PR, an optimizer of expected UCB (Daulton et al., 2022).

Algorithm 5 Iterative Best Response-Fitness (Ibr-Fitness)

Input: Domain $\mathcal{X}$, fitness function $f : \mathcal{X} \to \mathbb{R}$, players $\mathcal{N}$, initial data $\mathcal{D}_0 = \{(x_i, y_i = f(x_i) + \epsilon)\}$, batch size $B > 0$.

$x_0^{br} \leftarrow$ random joint strategy, $x_0^{br} \in \mathcal{X}$
for iteration $k = 1, 2, \ldots, K$ **do**
  Randomly selected B players $\in \mathcal{N}$ generates BRs $\{x_{k,i}\}_{i=1}^{B}$ w.r.t. x_{k-1}^{br} based on $f(\cdot)$
  Obtain evaluations $y_{k,i} = f(x_{k,i}) + \epsilon_{k,i}, \quad \forall i = 1, \ldots, B$
  Update $\mathcal{D}_k \leftarrow \mathcal{D}_{k-1} \cup \{(x_{k,i}, y_{k,i})\}_{i=1}^{B}$
  Play $x_k^{br} \leftarrow \arg\max_{x_{k,i} \in \{x_{k,i}\}_{i=1}^{B}} y_{k,i}$
end for
return $x_K^\star \leftarrow \arg\max_{(x,y) \in \mathcal{D}_K} y$ ▷ Best-so-far

Algorithm 6 Random

Input: Domain $\mathcal{X}$, $f : \mathcal{X} \to \mathbb{R}$, initial data $\mathcal{D}_0 = \{(x_i, y_i = f(x_i) + \epsilon)\}$, batch size $B > 0$.

for iteration $k = 1, 2, \ldots, K$ **do**
  Randomly generate batch of B points $\{x_{k,i}\}_{i=1}^{B}, \forall x_{k,i} \in \mathcal{X}$
  Obtain evaluations $y_{k,i} = f(x_{k,i}) + \epsilon_{k,i}, \quad \forall i = 1, \ldots, B$
  Update $\mathcal{D}_k \leftarrow \mathcal{D}_{k-1} \cup \{(x_{k,i}, y_{k,i})\}_{i=1}^{B}$
end for
return $x_K^\star \leftarrow \arg\max_{(x,y) \in \mathcal{D}_K} y$ ▷ Best-so-far

Appendix D. Experiment details

In all experiments, we use a GP surrogate with an RBF kernel for GP-based methods. The RBF specifies lenghtscales for each input variable separately – sometimes known as ARD kernels Rasmussen et al. (2006). To handle categorical inputs to the GP surrogate, we employ feature embeddings as representations for these inputs. The prior mean for the GP is predefined as the average log fitness value over the whole dataset. Kernel hyperparameters are optimized prior to the start of optimization and remain fixed throughout the BO iterations; specifically, lengthscales are optimized over the training set at the start of each replication using Bayesian evidence, and the outputscale is fixed to the difference between the maximum fitness value observed in the dataset & mean. In other words, we fit also a prior mean. A consistent observation noise of 0.0004 is maintained for each training example. Detailed (hyper)parameters for the experiments can be found in Table 1.

To extract feature embeddings for $GB1(55)$ dataset, we use a pre-trained transformer protein language model from `esm` library by Meier et al. (2021).

GB1(4) The dataset (Wu et al., 2016) is fully combinatorial *i.e.* encompassing fitness measurements of 20^4 variants with 4 sites. In this context, each protein site is treated as a player in the cooperative game of GameOpt, with $\mathcal{N} = \{1, \ldots, 4\}$. Additionally, we also analyzed the effect of player grouping inspired by *epistasis* phenomenon in protein design and provided the analysis in Appendix E.

We train the GP surrogate by utilizing a small portion of the dataset, specifically 0.0625%, consisting of 100 protein variants. Since existing literature does not provide common ground

Table 1: (Hyper) parameter values.

(Hyper) parameter	Explanation	Value
K	The number of active learning (BayesOpt) iterations	50
T	The number of game rounds	400 for $GB1(4)$ and 200 for $GB1(55)$
$\mid \mathcal{N} \mid$	The number of players	4 for $GB1(4)$, 10 and 55 for $GB1(55)$
$\mid \mathcal{D}_0 \mid$	The number of samples in training set	100 for $GB1(4)$ and 1000 for $GB1(55)$
η	Learning rate	0.9
ϵ	Observation noise for each training example	0.0004
l	RBF kernel lengthscale	optimized offline
β	The UCB tuning parameter	2
B	Batch size per BO iteration	5

feature embeddings as representations for the $GB1(4)$ variants, we use chemical descriptors (Wu et al., 2019) to extract 60 feature embeddings using a training set of size 1000 protein variants with LASSO method. We apply k-fold cross-validation with $k = 18$ different train/test dataset partitions. Following this, we evaluate the performance of our approach over 18 replications. In each replication, we initialize the GP surrogate-based baseline methods with the same initial GP model as our approach. We also use the same initial protein sequence for comparison within that replicate but employ different initial points across replications. We set the starting joint strategy as the protein sequence having the highest log fitness value in the training set. The prior mean of the GP is fixed at 1.0162. For the kernel hyperparameters, 60 lengthscales are defined for each feature dimension and optimized offline at the beginning of a replication; outputscale is set to 0.02169.

GB1(55) We experiment on the non-exhaustive dataset $GB1(55)$ that only includes 2-point mutations throughout the entire 55 residues of the $GB1$ protein resulting in $535,917$ variants (Olson et al., 2014) and consider two settings: 55 and 10 number of players.

GB1(55) with 55 Players In this context, we treat each protein site as a player in the GameOpt, thus, $\mathcal{N} = \{1, \ldots, 55\}$.

As the dataset is not completely combinatorial, we do not have access to measured fitness values for all 20^{55} variants. To overcome this, we employ a Deep Neural Network-based (DNN) *oracle* to predict fitness scores using feature embeddings associated with the protein sequences. We again opt to feature embeddings as the representation for categorical input of GP surrogate. Unlike $GB1(4)$, we utilize the `ESM-1v` protein language model from `esm` introduced by Meier et al. (2021), specifically designed for predicting protein variant effects and can be used to extract embeddings. With `ESM-1v`, we represent a sequence through a 1280 dimensional feature embedding vector. We train the *oracle* with supervised learning, using the training set having $(477\,854 \times 1280, 477\,854)$ feature & label pairs. Obtaining the exhaustive version of the $GB1(55)$ dataset, we train the GP surrogate using `ESM-1v` feature embeddings of 1000 randomly generated protein variants and corresponding *oracle*-predicted fitness scores for 10 replications.

GB1(55) with 10 players To further analyze the performance of GameOpt compared to the other baselines, we consider the setting where among the 55 sites, only 10 most significant protein sites can be mutated.

We employ the same protein language model for embeddings and *oracle* to predict fitness scores. However, the choice of 10 players among $\binom{55}{10}$ possibilities is a strategic decision

that affects the design performance. For this, we define the significance of a protein site considering the average variation in the fitness scores in the dataset. Concretely, we use Algorithm 7 and select $\mathcal{N} = \{21, 24, 35, 39, 41, 45, 46, 47, 48, 50\}$ sites as the players. We treat the rest of the protein sequence *i.e.* sites that do not correspond to players as fixed.

Algorithm 7 ComputeMostSignificantSites

Input: Dataset $\mathcal{D} = (x_i, y_i)_{i=1}^N$, players K, protein sequence length L, amino acids set $\mathcal{A}$.

Initialize $players \leftarrow \emptyset$, $site\_score_a^k \leftarrow \emptyset$ and $site\_var^k \leftarrow 0, \forall k \in \{1, \ldots, L\}, a \in \mathcal{A}$
for each pair $(x_i, y_i) \in \mathcal{D}$ **do**
    for each site $k \in \{1, \ldots, L\}$ **do**
        Set amino acid in site k as $a \leftarrow x_i^k$
        Append $site\_score_a^k \leftarrow site\_score_a^k \cup \{y_i\}$
    end for
end for
$site\_score^k \leftarrow \bigcup_{a \in \mathcal{A}} site\_score_a^k, \forall k \in \{1, \ldots, L\}$
$site\_var^k \leftarrow stdev(site\_score^k), \forall k \in \{1, \ldots, L\}$
return K sites having highest $site\_score$ as $players$

Appendix E. Experiment results

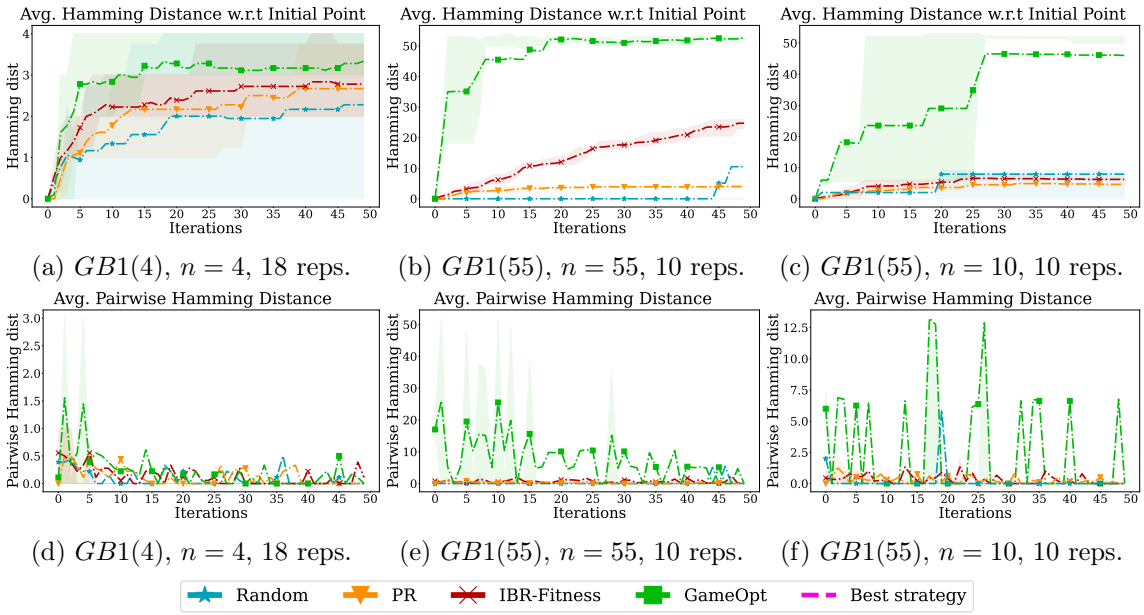
We also evaluate the performance of methods in terms of sampled batch diversity measured via the mean Hamming distance of sampled variants to the (1) closest initial training point and (2) proposed variant at the previous iteration (pairwise distance).

From Figure 3, it is clear that GameOpt consistently samples diverse evaluation points both with respect to the initial closest point from the training set and to the previously executed strategy in comparison to the other baselines. Diversity in the input space enables GameOpt to explore effectively and discover informative evaluation points.

IBR-Fitness shows a moderate sample diversity which might be due to its more exploitative behavior in relation to true log fitness values compared to other baselines. However, its exploration comes from the randomly generated B best responses, which may not always guarantee a diverse sample. To this end, IBR-Fitness may not overcome the challenges addressed above. Although PR maintains a diverse batch in $GB1(4)$, it fails to show the same performance for the other settings. Furthermore, to sample a batch, PR needs to compute the expected UCB over all possible strategy combinations of players, making its performance and hence sample diversity highly dependent on an accurate estimate of this expectation. Finally, Random exhibits poor performance across all experiments.

E.1 Players' Grouping

In light of the *epistasis* (Phillips, 2008) phenomenon in protein design, which underscores how the effect of a mutation on fitness can be influenced by the presence of other mutations within the same protein, we explore the concept of grouping protein sites together, *i.e.* having players being responsible for more than one site. This is because modeling protein sites independently may yield different fitness outcomes than finding equilibria among groups



(a) $GB1(4)$, $n = 4$, 18 reps. (b) $GB1(55)$, $n = 55$, 10 reps. (c) $GB1(55)$, $n = 10$, 10 reps.

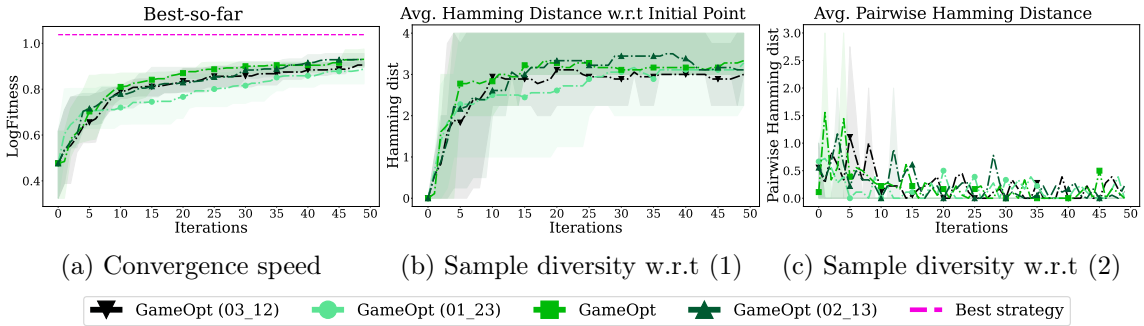
(d) $GB1(4)$, $n = 4$, 18 reps. (e) $GB1(55)$, $n = 55$, 10 reps. (f) $GB1(55)$, $n = 10$, 10 reps.

Figure 3: Sample diversity of methods measured via mean Hamming distance of sampled variants w.r.t. (1) the closest initial point from the training set and (2) proposed variant at the previous iteration (pairwise distance), under batch size $B = 5$. Each point on each line is the average of multiple replications initiated with different training sets having 100 and 1000 variants for $GB1(4)$ and $GB1(55)$. Similarly, error bars are interquartile ranges averaged over replications. In all experiments GameOpt samples a much diverse batch of evaluation points w.r.t. both measures—resulting in an outperforming performance compared to baselines.

of several sites. To this end, we conduct a preliminary investigation into whether this phenomenon alters GameOpt's performance.

We experiment on $GB1(4)$ with $\{0, 1, 2, 3\}$ protein sites and $\mathcal{N} = \{1, 2\}$ players, considering 3 possible player & site groupings: $\{(01, 23), (02, 13), (03, 12)\}$. For instance, setting $(01, 23)$ means that the first player is responsible for sites $\{0, 1\}$ and the other one for $\{2, 3\}$.

Our evaluations using the same performance measures (Figure 4) showed that there is no significant performance difference between individual players and grouping settings as they all discover the high log fitness valued protein variants at a similar rate while collecting batches of diverse evaluation points.



(a) Convergence speed (b) Sample diversity w.r.t (1) (c) Sample diversity w.r.t (2)

Figure 4: GameOpt performance for player grouping, under $GB1(4)$ setting, 18 reps.