BAOSA: Benchmarking Active Optimization for Self-driving lAboratories

Anonymous authors
Paper under double-blind review

Abstract

Discovery of novel materials and antibiotics can be posed as an optimization problem, namely, identifying candidate formulations that maximize one or more desired properties. In practice, however, the enormous dimensionality of the design space and the high cost of each experimental evaluation make exhaustive search strategies infeasible. Active learning methods, which iteratively identify informative data points, offer a promising solution to tackle these challenges by significantly reducing the data-labeling effort and resource requirements. Integrating active learning into optimization workflows, hereafter termed active optimization, accelerates the discovery of optimal candidates while substantially cutting the number of required evaluations. Despite these advances, the absence of standardized benchmarks impedes objective comparison of methodologies, slowing progress in self-driving scientific discovery. To address this, we introduce BAOSA, a comprehensive benchmark designed to systematically evaluate active optimization in self-driving laboratories. BAOSA provides a standardized evaluation protocol and reference implementations to facilitate efficient and reproducible benchmarking. BAOSA includes both synthetic benchmarks and real-world tasks in various fields, designed to address unique challenges, particularly limited data availability, in self-driving laboratories.

1 Introduction

Optimizing proteins and materials to achieve targeted properties, from antibiotic efficacy to superconductivity, represents a crucial frontier to address critical scientific and societal challenges (Hamidieh, 2018; Varmus et al., 2003; Merchant et al., 2023). Traditionally, scientists have approached these design processes by generating hypotheses based on prior knowledge and past data. These hypotheses are then tested using experimental protocols within constrained budgets. However, this approach is often inefficient, time-consuming, and limited by human ingenuity and errors. In recent years, the integration of data-driven methods with real-world laboratory setups has accelerated discovery across various fields, ranging from the design of proteins and DNA sequences in biology to the discovery of functional materials (Coley et al., 2019; Rao et al., 2022; Szymanski et al., 2023; Rapp et al., 2024).

One of the most promising innovations in this field is self-driving laboratories (SLs), which leverage active learning (AL) algorithms to autonomously guide experimentation and accelerate scientific discovery (Häse et al., 2019; Kang et al., 2019; Abolhasani & Kumacheva, 2023). AL is a paradigm in which a learning algorithm iteratively selects the most informative unlabeled samples to be labeled by an oracle (e.g., an experiment or human expert), with the goal of achieving higher model accuracy using fewer labeled data points compared to passive (random) sampling (Ren et al., 2021). Advances in AL offer the potential to significantly enhance the exploration of larger regions within the expansive search space, thus improving efficiency and effectiveness in experimental designs and optimization processes. As the central aim of SLs is optimization, we refer to the entire pipeline as active optimization (AO).

AO shares the goals and overall workflow of Bayesian optimization (BO) (Shahriari et al., 2016; Bubeck et al., 2011; Springenberg et al., 2016; Garnett, 2023), but broadens its methodological scope. While classical BO often employs kernel-based surrogate models paired with uncertainty-driven acquisition functions to propose new candidates, AO can in principle integrate any surrogate model or search strategy (including BO itself, by performing BO surrogate training and sampling on the resulting model), making it more versatile across problem domains. Likewise, AO resembles AL in its iterative sample selection, yet differs in its ultimate aim: rather than improving predictive accuracy of the surrogate model, AO focuses on locating optimal solutions from the given dataset. AO comprises two core components: (1) a surrogate model that approximates the complex mapping from design parameters to target properties; and (2) a search strategy that leverages this surrogate to iteratively generate and evaluate candidates, guiding the search toward the optimal solution. The complete AO pipeline is illustrated in Figure 1(a).

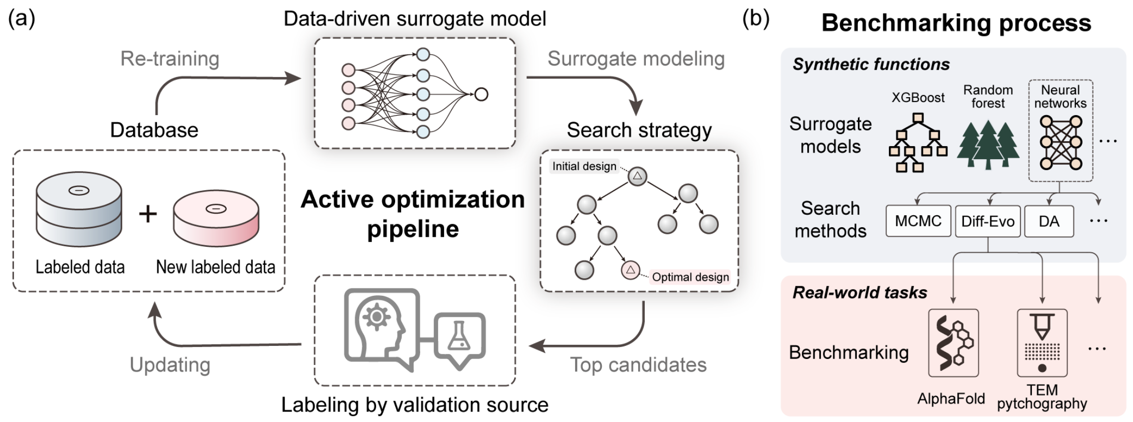

Figure 1: Overview of key components in active optimization for self-driving labs. (a) The goal of the active optimization task is to iteratively and autonomously find the optimal solutions. Beyond synthetic functions, the proposed BAOSA utilizes i) AlphaFold2 as a simulator for biology applications and ii) open-source scanning transmission electron microscopes (STEM) simulators for materials science applications. (b) Various surrogate models and search methods are first evaluated on synthetic functions, then the top-performing search strategies are applied to real-world tasks.

Despite significant progress, the search strategies used in AO, often fail to adapt and scale effectively to the high-dimensional, non-linear landscapes characteristic of many scientific applications (Frazier, 2018). BO and its variants that learns a probabilistic surrogate of the objective function and sample the best candidates using an uncertainty-based technique such as Thompson sampling (Shahriari et al.), perform well in low-dimensional spaces, but deteriorate in performance as problem dimensionality and complexity increase (Frazier, 2018). More recently, tree search methods, which are the key component of many revolutionary AI algorithms such as AlphaGo (Silver et al., 2016), have been applied to design problems. These methods iteratively partition the search space (Kim et al., 2020a) and employ local surrogate models to approximate the promising search subspace (Eriksson et al., 2019). However, their success is often contingent on the quality of these local models, and they also struggle with the curse of dimensionality (Wang et al., 2020).

Moreover, the intricate interplay between surrogate models and search strategies within AO pipelines, coupled with the growing number of scientific applications, has made it increasingly difficult to compare and track progress effectively. Different methods are often proposed and evaluated on distinct tasks with varying evaluation protocols, leading to inconsistent benchmarks. To the best of our knowledge, no unified benchmark or systematic investigation currently exists to evaluate and compare different search strategies in AO pipelines. This paper addresses this gap by proposing a standardized platform that enables a fair comparison of state-of-the-art search strategies, ensuring more consistent progress in scientific discovery. [1]

Our main contributions are summarized as follows:

- We introduce BAOSA, a platform tailored for evaluating search strategies within active optimization pipelines in real-world self-driving settings, emphasizing the employment of iterative process, surrogate models in low-data regimes.

[1]The code and benchmark suite for BAOSA are publicly available at `https://github.com/anonymized`.

- We provide a suite of 8 standardized synthetic tasks and 14 baseline methods, and 4 real-world tasks, covering various fields, for systematic and comprehensive evaluation.
- We perform large-scale benchmark studies to highlight critical areas for advancing self-driving labs within the AO pipeline: (i) understanding the interplay between surrogate model and search strategy in relation to the objective landscape, (ii) ensuring reproducibility of algorithmic performance across a wide variety of synthetic and real-world tasks with limited data availability.

1.1 Related work

Self-driving laboratories There has been a surge of interest in developing SLs across numerous scientific domains. For example, AL-driven pipelines have been applied to organic small-molecule synthesis and compound discovery (Li et al., 2015; Coley et al., 2019), synthetic biology platforms (Martin et al., 2023), and closed-loop drug discovery workflows (Saikin et al., 2019). In chemistry, active learning and optimization methods have enabled multi-step reaction planning (Epps et al., 2020; Seifrid et al., 2022; Boiko et al., 2023; Volk et al., 2023), reaction-condition screening (Torres et al., 2022; Angello et al., 2022), copolymer design (Reis et al., 2021), and fully autonomous chemical synthesis (Manzano et al., 2022). In materials science, similar approaches have accelerated the discovery of solid-state materials (Szymanski et al., 2023; Merchant et al., 2023) and clean-energy compounds (Tabor et al., 2018), as well as the optimization of thin-film fabrication processes (Ludwig, 2019). Each of these studies implements an AO pipeline tailored to its specific experimental constraints and objectives. Due to the rapid pace of development and interest across various disciplines, we can only include a limited selection. A curated and up-to-date list across application areas and a broad overview of SLs including applications, software packages, or hardware is provided by the Canadian Acceleration Consortium (Consortium).

Benchmarks Different platforms and benchmarks have been proposed for black-box optimization. COCO (Hansen et al., 2021) is a platform for comparing continuous optimizers in a black-box setting. Design-Bench Trabucco et al. (2022) proposed a benchmark for offline model-based optimization. Further benchmarks include robotics systems (Ginsburg et al., 2023) or simple multi-tool motion platforms (jub). Other works developed codebases for optimization algorithms and libraries without downstream tasks or datasets (Rapin & Teytaud, 2018). However, all of these benchmarks primarily measure performance by the number of function evaluations needed to reach a global optimum, often focusing on trajectory planning scenarios. This paradigm is ill-suited to real-world self-driving laboratories, where direct evaluations of the true objective can incur prohibitive costs.

2 BAOSA: Proposed platform

BAOSA is a comprehensive benchmark platform that can evaluate different search strategies in active optimization for real-world SLs, which acts as a critical step before applying these strategies to experimental laboratory environments. BAOSA is specifically designed for self-driving laboratories, where obtaining exact measurements can be prohibitively expensive. In such settings, surrogate models are used to approximate the true objective function, and different search strategies are applied to these surrogates rather than directly to the exact function. This differs from traditional optimization benchmarks, which typically assume direct access to the true function. This constraint simulates a common challenge in scientific problems, where data labeling is expensive and time-consuming in real-world scenarios.

Figure 1 (a) illustrates the general protocol of an AO pipeline, which comprises four main steps: (i) database, (ii) surrogate model that accurately represents complex relationships in the data, (iii) search strategy that utilizes the surrogate model to guide the search for an optimal single state, and (iv) validation source that can provide the ground truth. Although BAOSA follows this AO pipeline to evaluate both synthetic and real-world tasks, we intentionally add constraints to the number of data points in the database to emulate real-world environments. As a result, surrogate model training will be less effective, leading to more demanding scenarios for search strategies to explore optimal designs.

We include both standardized synthetic functions and real-world tasks to systematically evaluate a broad range of current search strategies and the respective surrogate models. The procedure of our benchmark is shown in Figure 1(b), where we first evaluate and compare a suite of surrogate models and search methods on synthetic test functions, and then apply the top-performing methods to a diverse set of real-world optimization tasks. Notably, we design and implement four real-world tasks to evaluate the proposed AO pipelines: (i) neural network architecture search to optimize model performance; (ii) the lunar landing problem, simulating complex control dynamics; (iii) a biology task utilizing AlphaFold2 as a virtual simulator for protein design, demonstrating applications in computational biology; and (iv) a materials science task focused on resolution optimization of scanning transmission electron microscopes, leveraging professional open-source simulation software for advanced imaging applications.

3 Problem statement

The goal of SL tasks is to discover the most informative data points while keeping labeling costs low. Each task is equipped with a quantitative metric that scores candidate designs, and our objective is to find the global optimal (minimizer as an example) of this metric:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in X} f(\mathbf{x}) \tag{1}$$

where $\mathbf{x}$ is the input vector and X is defined as the search space, typically $\mathbb{R}^n$, and n is the dimension. f is a deterministic but expensive objective function that maps the input $\mathbf{x}$ to the label. Because querying f directly is costly, we instead learn a data-driven surrogate model $\hat{f}$ from the dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_i^N$, in which N is the number of labels and y_i is the label of $\mathbf{x}_i$.

The AO pipeline involves iteratively selecting the samples $\mathbf{x}_{\text{new}}$ from X, based on a search strategy Q defined on the surrogate model $\hat{f}$, and re-training the surrogate model $\hat{f}$ with the new labeled data $(\mathbf{x}_{\text{new}}, y_{\text{new}})$. y_{new} is stored at each iteration as the optimization history to track the performance of Q. Each iteration t consists of three steps:

1. Surrogate Training: Train the surrogate model $\hat{f}_{\theta_t}$ using the current labeled dataset $\mathcal{D}$:

$$\theta_t = \arg\min_{\theta} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[L(f_\theta(\hat{\mathbf{x}}), y)]$$

   where L is the loss function.

2. Search and Selection: Using a search strategy applied to $\hat{f}_{\theta_t}$, choose a batch of top k promising, unlabeled points from optimization history:

$$\mathbf{x}_{\text{new}} = Q(\hat{f}_{\theta_t}; k) \in X$$

3. Labeling and Updating: Obtain the labels y_{new} for the selected samples $\mathbf{x}_{\text{new}}$ from the exact function f, add them to the labeled dataset $\mathcal{D}$:

$$y_{\text{new}} = f(\mathbf{x}_{\text{new}}), \quad \mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_{\text{new}}, y_{\text{new}})\}$$

The use of batched data is proposed to accommodate parallel experiments commonly encountered in real-world problems. It is noteworthy that this function is not limited to single-objective problems; it can be a product of multiple functions as long as it solely depends on $\mathbf{x}$, which makes it a multi-objective task.

4 Synthetic benchmark tasks

We conducted comprehensive benchmark on eight carefully selected functions: Ackley, Rastrigin, Rosenbrock, Griewank, Schwefel, Michalewicz, Levy, and Sphere. The primary objective for these synthetic functions is to identify their global minima with a minimum number of sample acquisitions. Unlike traditional optimization algorithms, which primarily focus on minimizing the number of function evaluations required to reach the global optimum, our benchmark study uses these synthetic functions to mimic the complex data distributions

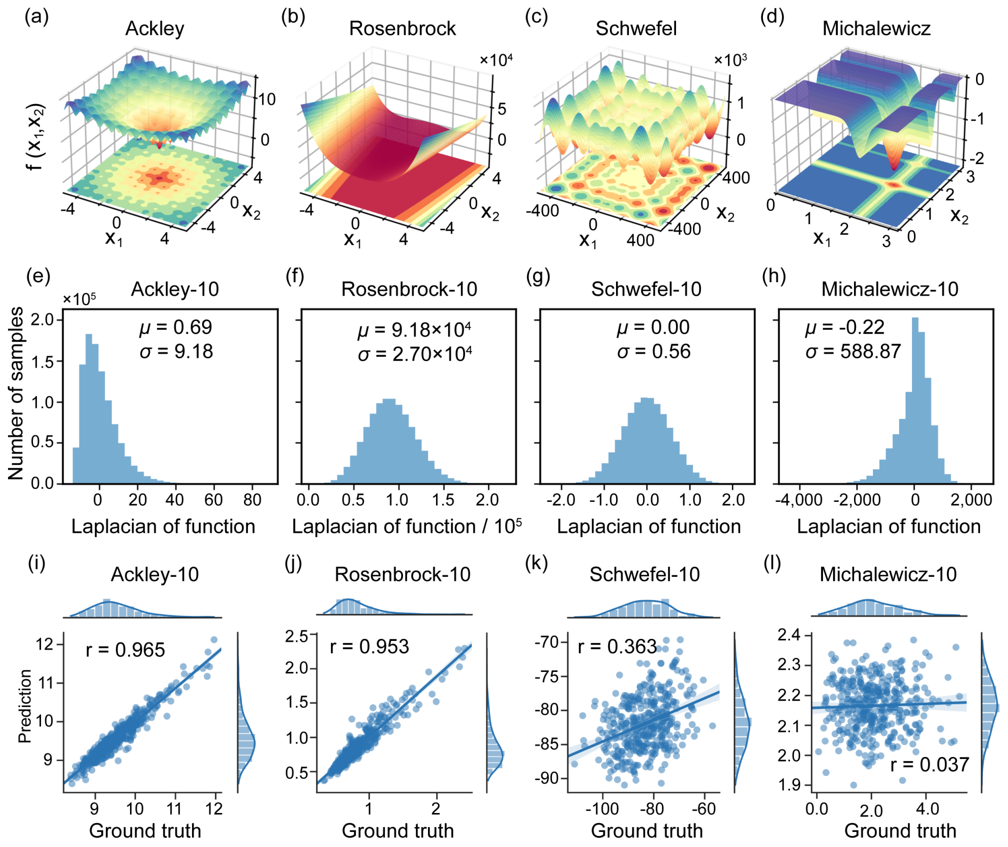

Figure 2: Objective landscapes of different synthetic functions with distinct topological characteristics. Visualization of 2D objective landscapes with (a) Ackley, (b) Rosenbrock, (c) Schwefel, and (d) Michalewicz in their 2D forms. Histograms (frequency distributions) of Laplacian of function s for (e) Ackley, (f) Rosenbrock, (g) Schwefel, and (h) Michalewicz, where each function is in 10-dimension with 1 million samples uniformly sampled from the parameter space. Joint plots of the ground truth function values (x-axis) and the surrogate model predictions (y-axis) for (i) Ackley, (j) Rosenbrock, (k) Schwefel, and (l) Michalewicz, where r denotes the Pearson correlation coefficient. Note that some of the functions are re-scaled to achieve better fitting (see Appendix A.4 for more details.)

generated by various validation sources. The process is iterative, with each iteration allowing only 20 data points to be sampled from the synthetic function tasks. This constraint necessitates the development of an effective learning-based surrogate model. To better mimic real experimental uncertainty, relative Gaussian noise can be added to the synthetic functions. These synthetic functions can serve as valuable test cases for understanding the properties of real-world SL tasks across diverse conditions using different search algorithms with surrogate models within the AO pipeline. Given the result in synthetic functions, we will further benchmark the methods that performed well on real-world tasks. First, we explore a statistical feature to characterize different objective landscapes that may pose challenges for the AO search strategies. Here, we focus on four key functions: Ackley, Rosenbrock, Schwefel, and Michalewicz, as these functions are characterized by their distinct objective landscapes.

Landscape characterization Landscape topology is a critical feature of an objective function. Previously, exploratory landscape analysis (ELA) have quantified it using metrics such as the probability of convexity or the estimated number of local optima (Mersmann et al., 2011). A machine learning model often exhibits a less-satisfactory performance on a flat landscape of an objective function, for which most of the values are at the same level, making it difficult for the model to learn and generalize. A poorly performing surrogate model may mislead the search methods, ultimately resulting in sub-optimal outcomes. Figure 2 (a-d) visualizes the

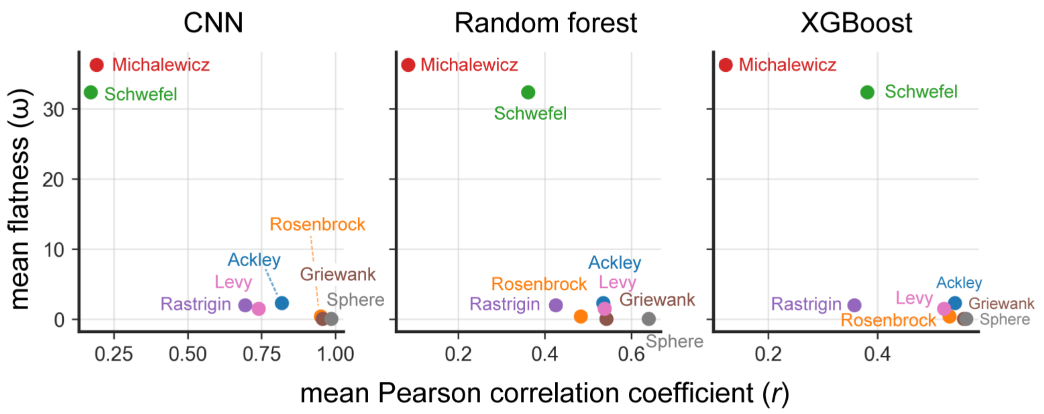

Figure 3: Correlations between Pearson correlation coefficient (r) and flatness (ω) for CNN, Random Forest, and XGBoost across benchmark functions. Note that all functions are in 50D.

objective landscapes of the corresponding synthetic functions in their 2D forms. Ackley shows a rugged but funneled topology, while Rosenbrock exhibits a long valley with numerous local minima. Schwefel presents a complex multi-funnel topology, whereas Michalewicz has sharp drops on a rather flat landscape (The mathematical formula can be found in Appendix A.1).

To better understand the relationship between the landscape of the objective function and the performance of the surrogate model, we introduce the landscape flatness. This metric uses random sampling and discrete Laplacian operator to quantify the flatness of the objective landscape. While the metric provides valuable empirical insights, we acknowledge its limitations in theoretical rigor and aim to explore a more comprehensive analysis in future work.

Laplacian of function Let $\mathbf{x} = [x_1, ..., x_i, ..., x_n]$ be a n-dimensional input of the function. The discrete Laplacian operator at a high-dimensional position $\mathbf{x}$ can be defined as:

$$s_x = \sum_{i=1}^{n} \frac{\partial^2 f}{\partial x_i^2} \approx \sum_{i=1}^{n} \frac{f(x_i + \epsilon) + f(x_i - \epsilon) - 2f(x_i)}{\epsilon^2} \tag{2}$$

where ϵ is the step size and is set to 0.01 partition of the interval between upper bound and lower bound. The Laplacian of function s is expected to be positive for a locally convex landscape in many of the i^{th} dimensions and to be negative for a locally concave landscape in many of the i^{th} dimensions. A near-zero Laplacian of the function s indicates that the objective function has a rather flat distribution, and there is no gradient on the landscape in many of the i^{th} dimensions.

Figure 2 (e-h) demonstrate the frequency distributions of s and the corresponding mean μ and standard deviation σ, where we uniformly sampled 1 million inputs from the individual parameter spaces (in 10D) of the functions. Ackley shows a positively skewed distribution with μ close to 0 and σ of 9.18, suggesting a moderate fluctuation in concavity across all dimensions with some more convex areas (Figure 2e). Rosenbrock shows both large μ of 9.18×10^4 and σ of 2.70×10^4, indicating a landscape that is heavily convex anywhere in the landscape domain, with highly anisotropic concavity across all dimensions (Figure 2f). In contrast, Schwefel shows near-zero values for both μ and σ, implying a landscape that is generally flat with a rather small, isotropic concavity across all dimensions (Figure 2g). Interestingly, Michalewicz shows a μ close to 0 and an abnormally large σ, implying that the landscape is flat with some small areas being dramatically concave or convex (Figure 2h).

Landscape flatness To enable a dimensionless comparison of landscape flatness across different function forms, we adapt the ELA curvature feature into a new metric ω. Denoting by μ and σ the mean and standard deviation of the frequency distributions of s, we define

$$\omega = \sqrt{\frac{\sigma}{|\mu|}}.$$

Ackely-10 and Rosenbrock-10 have ω of 3.62 and 0.54, respectively, whereas ω of Schwefel-10 and Michalewicz-10 are 37.07 and 54.47, respectively, indicating that the overall landscape is rather flat. Figure 3 demonstrates the relationship between the flatness measure ω and surrogate performance across model types (Random Forest (Breiman, 2001) and XGBoost (Chen & Guestrin, 2016)). Appendix Figure A5 extends this analysis across multiple input dimensionalities. We observe that at low dimensionalities (e.g., 10D), tree-based surrogates match or even surpass the 1D-CNN's fit. However, their performance degrades sharply as dimensionality increases. In contrast, the 1D-CNN maintains a stable fit up to 100D, except for functions with high flatness ω, and continues to outperform tree-based models at 200D for all benchmark functions. Based on the experiments, we adopt a neural-network-based surrogate in our pipeline. In particular, functions with lower ω values are consistently easier for surrogate models to learn than those with higher ω values.

Surrogate model training A key challenge for AO search strategies with surrogate models is to learn a good approximator of the objective function with only a few samples. Figure 2 (i-l) presents the correlations of the ground truth function values and the surrogate model (i.e. neural network in this case) predictions for different functions (all in their 10D forms). Each surrogate model $\hat{f}$ was trained on a dataset $\mathcal{D} = \{(\mathbf{x}_i, f(\mathbf{x}_i)\}$ of inputs $\mathbf{x}_i$ and the corresponding function value $f(\mathbf{x}_i)$ (see Appendix A.4 for more details). It can be observed that surrogate models generalize better on landscapes with gradients (i.e., Ackley and Rosenbrock), and worse on flat landscapes (i.e., Schwefel and Michalewicz). It is likely that a surrogate model requires many more samples to generalize in the low ω scenario.

5 Benchmarking search strategy

With the surrogate model fixed as a 1D-CNN in the AO pipeline, we conducted a comprehensive benchmark of search strategies on both synthetic functions and carefully selected real-world tasks. In the current benchmark, we focus on four SL tasks: neural network architecture search (NAS), lunar landing problem, cyclic peptide design and optimization of electron ptychography reconstruction. These benchmark tasks are selected because (i) they are supported by accurate high-fidelity simulators, (ii) they address optimization problems with single or multiple objectives in the broad fields of computer science, automation control, biology, and materials science, and (iii) they can be executed within reasonable time and computational resources. Note that the experimental setups for NAS and lunar landing problem are detailed in Appendix A.6 and A.7, respectively.

5.1 Cyclic peptide design

Background Cyclic peptides are a class of compounds that have garnered significant attention as therapeutic agents due to their enhanced stability, high specificity, and excellent membrane permeability. These properties make them particularly effective in targeting traditionally "undruggable" protein surfaces (Vinogradov et al., 2019). The amino acids (AAs) in cyclic peptides are interconnected by amides or other chemically stable bonds, which can be chosen from the 20 standard AAs or various non-standard ones, creating a high-dimensional and complex sequence design space (Zorzi et al., 2017).

Here, the task is more specific than general protein design: it involves designing a specialized type of protein with therapeutic applications. This protein is required to exhibit stronger interactions with its target, such as higher binding affinity. Such a task can be framed as an optimization problem. However, even for a relatively simple 16-residue sequence, the combinatorial search space includes 16^{20} possible configurations. The intricate and nonlinear relationship between protein sequence and functional properties further complicates the challenge, making it a suitable benchmark for testing advanced methodologies. An additional advantage of this setup is the availability of natural binders as a reference for comparison. Traditionally, one often needs to conduct high-throughput wet lab experiments, synthesizing thousands of cyclic peptides before discovering one that can specifically bind to a desired protein (Gang et al., 2018). SL can accelerate this discovery process by narrowing the potential candidates to a few dozen, drastically reducing the cost. The general pipeline of this task is present in Figure 4 (a).

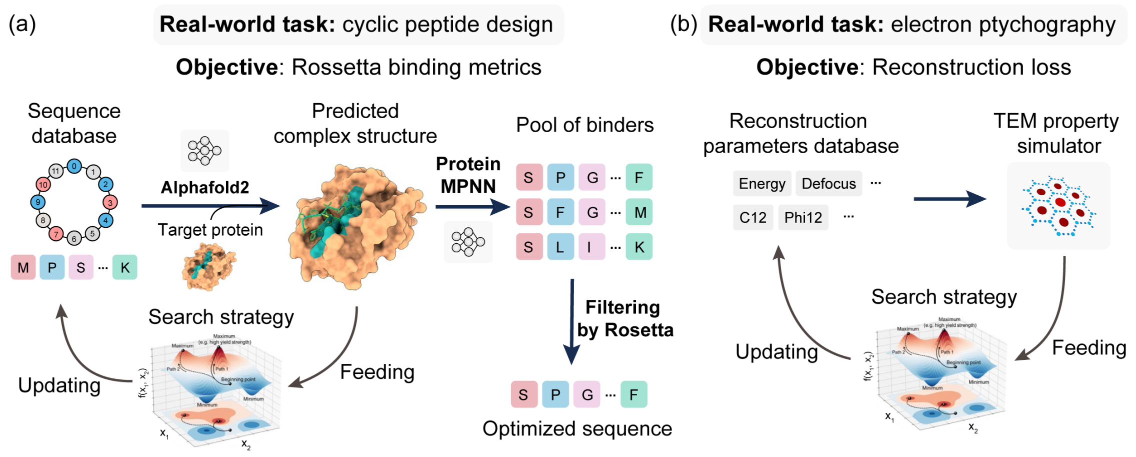

Figure 4: Pipelines of two chosen real-world tasks: (a) cyclic peptide design and (b) electron ptychography.

Dataset Two protein and canonical cyclic peptide complexes, PDBID: 4kel and PDBID: 7k2j, are sourced from the Protein Data Bank (PDB). The former is a 14-amino acid serine protease inhibitor targeting human kallikrein-related peptidase 4 (KLK4) (Riley et al., 2019), while the latter is a cyclic 7-mer peptide interacting with Kelch-like ECH-Associated Protein-1 (KEAP1) (Ortet et al., 2021). For simplicity, we only consider standard amino acids. Therefore, each cyclic peptide is represented as a sequence of integers ranging from 0 to 19, with each number corresponding to a distinct type of standard amino acid, making this a discrete optimization task.

Optimization target The optimization target of cyclic peptide design is defined as follows:

$$Target = SC \cdot dSASA \tag{3}$$

The SC value ranges from 0 to 1, referring to how well the surfaces of two proteins fit geometrically together at their interface; $dSASA$ measures the size of the interface (in units of Å^2). A larger $dSASA$ reflects a more extensive interface area. Further details regarding the dataset and simulation settings can be found in Appendix A.8.

5.2 Electron ptychography

Background Electron ptychography is a phase-contrast imaging technique capable of resolving nanostructures at a sub-angstrom resolution. Electron ptychography is widely used for specimens thicker than a monolayer (Cowley & Moodie, 1957) and sensitive materials vulnerable to beam-induced damage (Song et al., 2019). However, electron ptychography relies on a careful selection of various reconstruction parameters, such as physical, optimization, and experimental parameters, which affect the quality and accuracy of the retrieved transmission function. The parameter space is vast and complex, and the optimal choice depends on the specific configuration of the dataset and measurement conditions. Although some algorithms have been applied to this task (such as Bayesian optimization using Gaussian process (Cao et al., 2022)), the parameter selection process still strongly relies on expert knowledge and trial-and-error, which limits the efficiency and applicability of electron ptychography. The entire pipeline can be found in Figure 4 (b).

Dataset The dataset is a 4D datacube, comprising 2D grid of positions, each of which records a 2D diffraction pattern by a converged electron probe. Here, we utilized abTEM (Madsen & Susi, 2021) to simulate the dataset: 10-layer-stacked molybdenum disulfide (MoS_2), an emergent two-dimensional semiconductor that demonstrates strong potential to exceed the fundamental limits of silicon electronics (Li et al., 2024). The MoS_2 dataset is simulated with intentionally exaggerated probe aberrations to pose challenges for the optimization algorithms.

Optimization target The goal of this task is to optimize the reconstruction parameters within the electron ptychography algorithm to retrieve the best quality of phase of the transmission function within the atomic lattice. This requires solving a non-convex problem in a 15D parameter space in our case (see Appendix A.9 for details). Specifically, the objective function is the normalized mean square error (NMSE) between the positive square-root of the measured diffraction pattern I_M and the modulus of the Fourier-transformed simulated exit-wave Ψ, which can be formulated as:

$$\frac{1}{N}\sum_{i}^{N}\left|\sqrt{I_{M(i)}(\mathbf{u})} - |\mathcal{F}[\Psi_i(\mathbf{r})]|\right|^2 \tag{4}$$

where $\mathbf{r}$ and $\mathbf{u}$ denote the real- and reciprocal-space coordinate vectors, respectively, N is the total number of the measured diffraction patterns, and the operator $\mathcal{F}$ represents a Fourier transform. Further details regarding the dataset, simulation settings, and evaluation metrics can be found in Appendix A.9.

6 Results

6.1 Synthetic benchmarks

We benchmark 14 state-of-the-art search methods (including Random Search) alongside neural network as the surrogate model on synthetic function tasks within the AO pipeline. These methods span a wide range of algorithm categories, including Dual Annealing (DA (Pincus, 1970)), Evolutionary Algorithm (CMA-ES (Hansen et al., 2003), IPOP-CMA-ES (Auger & Hansen, 2005; Nomura & Shibata, 2024), BIPOP-CMA-ES (Hansen, 2009), Differential Evolution (Diff-Evo (Storn & Price, 1997)), Shiwa (Liu et al., 2020)), Bayesian Optimization (BO (Gardner et al., 2014), TuRBO (Eriksson et al., 2019), SAASBO (Eriksson & Jankowiak, 2021)), Monte Carlo Tree Search (LaMCTS(Wang et al., 2020), DOO (Munos, 2011), SOO (Munos, 2011), and VOO (Kim et al., 2020b)). The implementation settings of each AO search strategies can be found in Appendix A.5. Our evaluation covers all functions in their 20D forms, as well as the Ackley, Rastrigin, and Rosenbrock functions in both 20D and 100D forms. The full result can be found in Tabel 1. The history of the performance of AO search strategies for three selected functions can be found in Figure 5. Benchmark result and analysis of selected synthetic functions with different noise level and AO search strategies can be found in Appendix A.3.

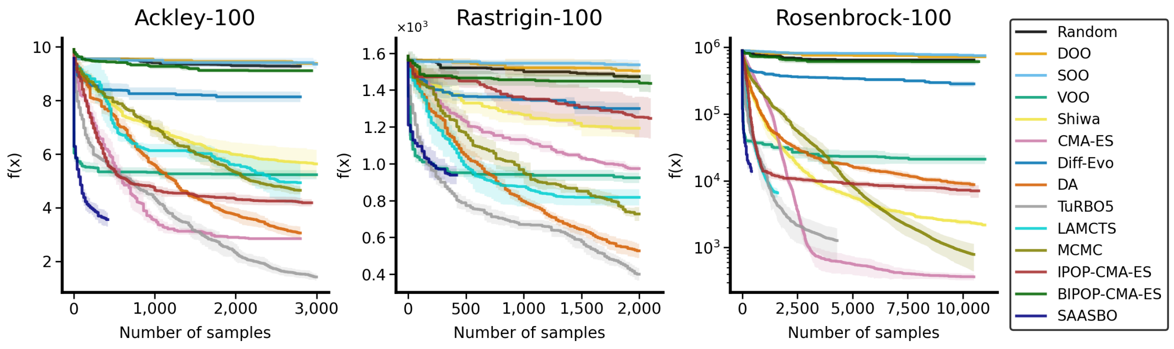

Figure 5: Evaluation of sampling efficiency for Ackley, Rastrigin, Rosenbrock in 100-dimension. SASSBO stops early because of out-of-memory. No single search strategy demonstrates consistent superiority across all scenarios.

Our benchmarking revealed a pronounced sensitivity to problem dimensionality: search strategies deliver strong performance in low-dimensional settings but deteriorate rapidly as dimensionality increases. We also found that their effectiveness hinges on the fidelity of the surrogate model—tasks with accurately fitted surrogates (e.g., Ackley and Rosenbrock) benefit substantially from advanced search, whereas functions with poor surrogate approximations (such as Schwefel and Michalewicz) show little to no advantage over random sampling, with only the 20-dimensional Ackley benchmark reliably reaching the global minimum.

Finally, although no single method dominates across all scenarios, TuRBO5 generally achieves the most robust performance across our suite of functions and dimensions. Although SAASBO excels under limited sampling budgets, it suffers from out-of-memory errors as the problem size grows.

The observed variance primarily arises from data sparsity associated with high dimensionality. Within our active optimization pipeline, we train a surrogate model that serves as the basis for exploration and optimization by search algorithms. Notably, the search algorithm operates without direct access to ground truth labels, making the random initialization of the surrogate model's training dataset a critical factor influencing the outcomes. Variations in these initializations yield distinct surrogate models, which in turn contribute to increased variance across trials. This effect is particularly pronounced in high-dimensional problems, where greater variance is anticipated due to the exacerbated sparsity.

Table 1: Evaluations of AO search strategies on synthetic functions with the usage of surrogate model, where the values with bold texts denote the best optimization result across all the methods. Results are averaged over 5 trials, and ± denotes the standard deviation.

	Ackley-20	**Ackley-100**	**Rastrigin-20** ($\times 10^2$)	**Rastrigin-100** ($\times 10^3$)	**Rosenbrock-20** ($\times 10^4$)	**Rosenbrock-100** ($\times 10^4$)	**Schwefel-20** ($\times 10^3$)	**Michalewicz-20**
$f(\mathbf{x}^*)$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-19.63
Random	7.60 ± 0.17	9.24 ± 0.13	2.15 ± 0.11	1.43 ± 0.016	2.370 ± 0.119	63.60 ± 0.936	5.50 ± 0.11	-6.11 ± 0.42
TuRBO5	0.37 ± 0.14	**1.73 ± 0.18**	**0.51 ± 0.03**	**0.41 ± 0.036**	**0.003 ± 0.001**	0.127 ± 0.066	2.84 ± 0.79	**-11.34 ± 1.20**
LaMCTS	1.95 ± 0.65	5.04 ± 0.53	0.83 ± 0.30	0.81 ± 0.034	0.007 ± 0.005	0.652 ± 0.098	3.32 ± 0.33	-7.56 ± 0.44
CMA-ES	0.65 ± 0.09	2.55 ± 0.04	0.76 ± 0.03	0.97 ± 0.017	0.005 ± 0.003	**0.036 ± 0.002**	5.28 ± 0.44	-6.38 ± 0.33
Diff-Evo	6.41 ± 0.15	8.03 ± 0.17	1.87 ± 0.12	1.31 ± 0.022	0.787 ± 0.115	28.20 ± 2.670	5.11 ± 0.16	-6.04 ± 0.63
DA	**0.00 ± 0.00**	3.18 ± 0.17	1.28 ± 0.06	0.51 ± 0.029	0.004 ± 0.003	0.911 ± 0.088	2.38 ± 0.39	-10.03 ± 0.77
Shiwa	4.41 ± 0.07	5.58 ± 0.52	2.18 ± 0.02	1.29 ± 0.047	2.216 ± 0.146	0.240 ± 0.022	5.41 ± 0.32	-6.35 ± 1.31
MCMC	0.005 ± 0.001	4.19 ± 0.06	0.59 ± 0.17	0.83 ± 0.058	0.021 ± 0.003	0.078 ± 0.038	**2.21 ± 0.83**	-9.64 ± 2.14
DOO	7.07 ± 0.35	9.43 ± 0.05	2.23 ± 0.34	1.54 ± 0.034	1.340 ± 0.326	78.22 ± 2.790	5.93 ± 0.21	-6.12 ± 0.38
SOO	7.65 ± 0.18	9.42 ± 0.15	2.21 ± 0.19	1.34 ± 0.087	2.750 ± 0.724	75.30 ± 2.910	2.39 ± 2.98	-6.44 ± 1.97
VOO	2.74 ± 0.39	5.93 ± 0.37	1.23 ± 0.53	0.91 ± 0.038	0.005 ± 0.000	2.101 ± 0.724	4.18 ± 0.18	-7.92 ± 0.89
IPOP-CMA-ES	2.23 ± 0.12	4.28 ± 0.15	1.14 ± 0.15	1.25 ± 0.109	0.010 ± 0.001	0.706 ± 0.150	4.58 ± 0.63	-6.62 ± 0.52
BIPOP-CMA-ES	6.91 ± 0.33	9.05 ± 0.03	1.77 ± 0.13	1.33 ± 0.038	2.795 ± 0.317	60.15 ± 4.248	5.51 ± 0.35	-6.28 ± 0.31
SAASBO	1.28 ± 0.35	3.34 ± 0.32	1.15 ± 0.23	0.91 ± 0.039	0.036 ± 0.005	1.084 ± 0.253	4.15 ± 0.43	-6.78 ± 0.28

All benchmark tasks here involve minimization objectives.
The asterisk (*) represents the global minimum of the function.

6.2 Real-world benchmarks

Drawing on our synthetic benchmark results, we selected some top-performing AO search strategies and evaluated them on real-world benchmark tasks under realistic cost constraints. For the NAS and lunar landing problem, we benchmark the results using six to nine different search strategies. For biology and materials science tasks, we evaluate the performance of four selected search strategies: Diff-Evo, DA, TuRBO5, and BO. Each task is subjected to three independent trials to ensure robust results, with each search strategy having a fixed number of oracle function evaluations.

Neural Architecture Search and Lunar Landing Problem Figure 6 shows benchmark results of both real-world problems. As for NAS, We benchmark the problem with six optimization algorithms: Random Search, MCMC, CMA-ES, DA, LAMCTS, and TuRBO5, where MCMC dominates and rapidly reaches 0.941 with 500 data acquisitions. Regarding the lunar landing, we evaluate this problem using nine algorithms: Random Search, DOO, SOO, VOO, Shiwa, CMA-ES, Diff-Evo, DA, and MCMC.

Cyclic peptide design

Electron ptychography Table 2 summarizes the performance of AO search strategies on ptychographic reconstructions of the MoS_2 dataset, where "Reference" denotes the expert reconstruction results for a single-layer MoS_2 dataset with the same aberration settings. It is observed that all AO search strategies do not achieve the optimal reconstruction of both object and probe functions. However, TuRBO5 and Diff-Evo can attain lower NMSE values and have generally more physically sensible reconstructions for the phases of the object transmission functions. As shown in Figure 8, despite not being perfect, both AO search strategies

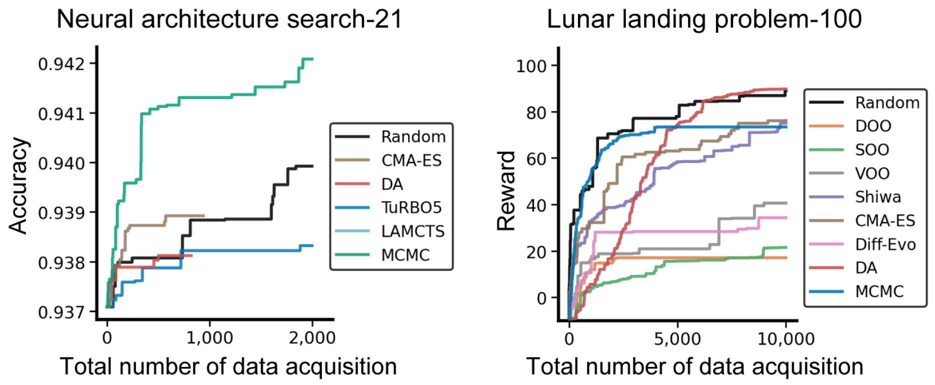

Figure 6: Benchmarks of sampling efficiency for Neural Architecture Search (NAS) in 21-dimension and lunar landing problem in 100-dimension. Note that both problems involve maximization objectives.

Table 2: Evaluations on two real-world tasks. Shape Complementarity (SC) and the change in Solvent Accessible Surface Area ($dSASA$) are used for cyclic peptide design, and normalized mean square error, object reconstruction or and probe reconstruction error are used for ptychographic reconstruction on the MoS_2 dataset. Upward arrow (↑) and downward arrow (↓) indicate maximization and minimization tasks, respectively. Results are averaged over 3 trials, and ± denotes the standard deviation.

	Cyclic peptide design						**Electron ptychography**		
	4kel-SC ↑	**4kel-dSASA ↑**	**4kel-Target ↑**	**7j2k-SC ↑**	**7j2k-dSASA ↑**	**7j2k-Target ↑**	**NMSE ↓**	**Object recon. error ↓**	**Probe recon. error ↓** ($\times 10^{-3}$)
Reference*	0.77	1505	1156	0.67	865	582	0.079	0.048	0.35
Diff-Evo	0.72 ± 0.05	1464 ± 65	1046 ± 69	0.66 ± 0.04	923 ± 72	613 ± 61	0.283 ± 0.005	0.102 ± 0.008	2.96 ± 0.34
DA	0.70 ± 0.03	1556 ± 32	1096 ± 48	0.65 ± 0.04	894 ± 59	570 ± 19	0.313 ± 0.005	0.118 ± 0.011	3.05 ± 0.27
TuRBO	0.71 ± 0.03	1501 ± 37	1059 ± 55	0.63 ± 0.01	904 ± 42	572 ± 17	0.275 ± 0.000	0.104 ± 0.001	2.60 ± 0.08
BO	0.72 ± 0.02	1431 ± 14	1035 ± 22	0.60 ± 0.03	908 ± 56	546 ± 57	0.300 ± 0.000	0.097 ± 0.000	3.28 ± 0.00

*Reference denotes "native" for cyclic peptide design and "expert reconstruction result" for electron ptychography.

(Diff-Evo and TuRBO5) can resolve the atomic contrasts of heavy Molybdenum (brighter) atoms and light Sulfur atoms (darker). On the other hand, DA and BO present higher NMSE values and are considered worse in ptychographic reconstruction. We note that although not able to fully resolve atomic contrasts from different atoms, BO has the lowest object reconstruction error and can retrieve an object transmission function with general atomic signals. More detailed analyses are included in Appendix A.9.

7 Discussion

Data-driven tasks in real-world problem represent exciting areas with tremendous potential for the development of self-driving labs. However, the absence of standardized benchmarks and evaluation protocols has hindered the accurate tracking of progress. To address this, we design an active optimization pipeline that tailors to self-driving lab settings, including (i) iterative process, (ii) use of surrogate models in low-data regime. Our platform BAOSA is a comprehensive resource that includes (i) a codebase for various search strategies, (ii) a suite of synthetic tasks, and (iii) two complex tasks with controlled simulators and two real-world applications in biology and materials science. It features a large-scale empirical evaluation and provides a template for reproducible research and for systematically advancing the performance of algorithms across disciplines, with virtual labs and high-fidelity simulators having the potential to reduce the need for costly and time-consuming real-world experiments. Our extensive benchmarks highlight current limitations and indicates promising directions for future research, including developing methods for hyperparameter selection with network-based surrogate models and scaling approaches to very high dimensions.

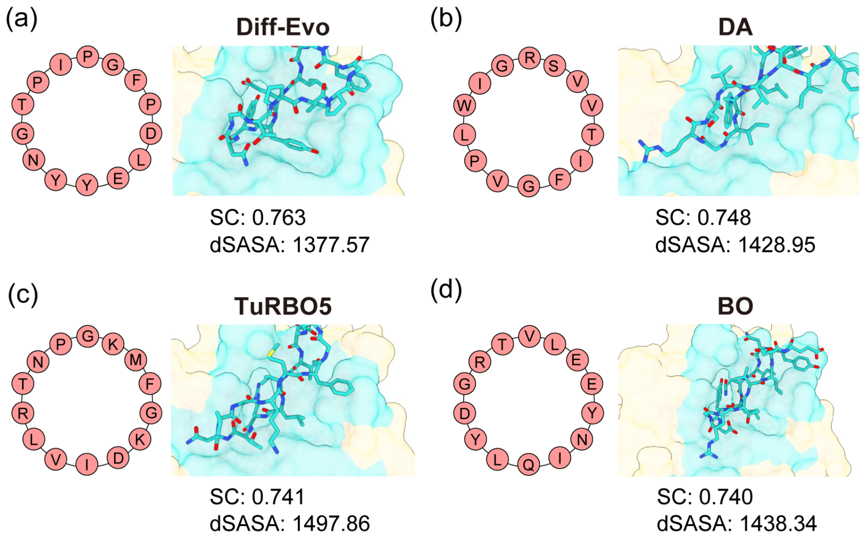

Figure 7: Benchmarking the cyclic peptide design task: visualization of protein 4kel yielded complex results, with the highest target value observed across three trials, where SC and dSASA denotes shape complementarity and change in Solvent Accessible Surface Area, respectively. The left inset illustrates the cyclic peptide sequence, while the right inset presents the interaction map for each method: (a) Diff-Evo, (b) DA, (c) TuRBO5, and (d) BO.

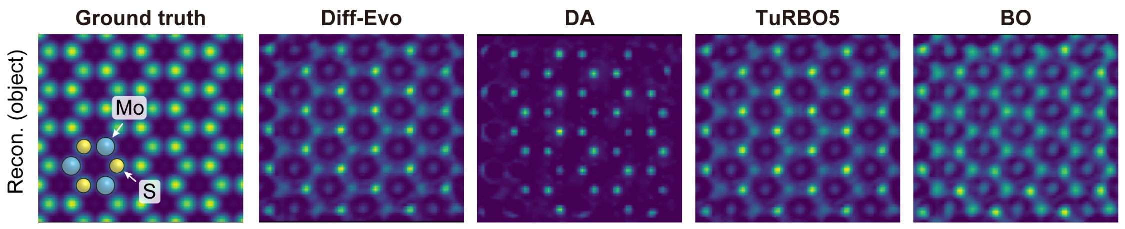

Figure 8: Benchmarking the electron ptychography task: visualization of the reconstructed phases (of the object transmission functions) with parameters obtained from the corresponding AO search strategies. No single strategy achieves results comparable to the ground truth.

References

Science Jubilee. https://github.com/machineagency/science-jubilee.

Milad Abolhasani and Eugenia Kumacheva. The rise of self-driving labs in chemical and materials sciences. *Nature Synthesis*, 2(6):483–492, 2023.

Nicholas H Angello, Vandana Rathore, Wiktor Beker, Agnieszka Wołos, Edward R Jira, Rafał Roszak, Tony C Wu, Charles M Schroeder, Alán Aspuru-Guzik, Bartosz A Grzybowski, et al. Closed-loop optimization of general reaction conditions for heteroaryl suzuki-miyaura coupling. *Science*, 378(6618):399–405, 2022.

A. Auger and N. Hansen. A restart cma evolution strategy with increasing population size. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pp. 1769–1776 Vol. 2, 2005. doi: 10.1109/CEC.2005.1554902.

Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in Neural Information Processing Systems 33*, 2020. URL http://arxiv.org/abs/1910.06403.

Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023.

Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvari. X-Armed Bandits. *Journal of Machine Learning Research*, 12:1655–1695, April 2011. URL https://hal.science/hal-00450235.

Michael C. Cao, Zhen Chen, Yi Jiang, and Yimo Han. Automatic parameter selection for electron ptychography via bayesian optimization. *Scientific Reports*, 12(1):12284, 2022.

Sidhartha Chaudhury, Sergey Lyskov, and Jeffrey J. Gray. PyRosetta: a script-based interface for implementing molecular modeling algorithms using Rosetta. *Bioinformatics*, 26(5):689–691, 01 2010. ISSN 1367-4803. doi: 10.1093/bioinformatics/btq007. URL https://doi.org/10.1093/bioinformatics/btq007.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Connor W Coley, Dale A Thomas III, Justin AM Lummiss, Jonathan N Jaworski, Christopher P Breen, Victor Schultz, Travis Hart, Joshua S Fishman, Luke Rogers, Hanyu Gao, et al. A robotic platform for flow synthesis of organic compounds informed by ai planning. *Science*, 365(6453):eaax1566, 2019.

Acceleration Consortium. Awesome Self-Driving Labs. https://github.com/AccelerationConsortium/awesome-self-driving-labs.

J. M. Cowley and A. F. Moodie. The scattering of electrons by atoms and crystals. i. a new theoretical approach. *Acta Crystallographica*, 10(10):609–619, 1957. doi: https://doi.org/10.1107/S0365110X57002194. URL https://onlinelibrary.wiley.com/doi/abs/10.1107/S0365110X57002194.

J. Dauparas, I. Anishchenko, N. Bennett, H. Bai, R. J. Ragotte, L. F. Milles, B. I. M. Wicky, A. Courbet, R. J. de Haas, N. Bethel, P. J. Y. Leung, T. F. Huddy, S. Pellock, D. Tischer, F. Chan, B. Koepnick, H. Nguyen, A. Kang, B. Sankaran, A. K. Bera, N. P. King, and D. Baker. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022. doi: 10.1126/science.add2187. URL https://www.science.org/doi/abs/10.1126/science.add2187.

Robert W Epps, Michael S Bowen, Amanda A Volk, Kameel Abdel-Latif, Suyong Han, Kristofer G Reyes, Aram Amassian, and Milad Abolhasani. Artificial chemist: an autonomous quantum dot synthesis bot. *Advanced Materials*, 32(30):2001626, 2020.

David Eriksson and Martin Jankowiak. High-dimensional Bayesian optimization with sparse axis-aligned subspaces. In Cassio de Campos and Marloes H. Maathuis (eds.), *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 493–503. PMLR, 27–30 Jul 2021. URL `https://proceedings.mlr.press/v161/eriksson21a.html`.

David Eriksson, Michael Pearce, Jacob R Gardner, Ryan Turner, and Matthias Poloczek. Scalable global optimization via local bayesian optimization. 10 2019. URL `http://arxiv.org/abs/1910.01739`.

Peter I. Frazier. A tutorial on bayesian optimization. 7 2018. URL `http://arxiv.org/abs/1807.02811`.

Donghyeok Gang, Do Wook Kim, and Hee-Sung Park. Cyclic peptides: Promising scaffolds for biopharmaceuticals. *Genes*, 9(11), 2018. ISSN 2073-4425. doi: 10.3390/genes9110557. URL `https://www.mdpi.com/2073-4425/9/11/557`.

Jacob R. Gardner, Matt J. Kusner, Zhixiang Xu, Kilian Q. Weinberger, and John P. Cunningham. Bayesian optimization with inequality constraints. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pp. II–937–II–945. JMLR.org, 2014.

Roman Garnett. *Bayesian Optimization*. Cambridge University Press, 2023.

Tobias Ginsburg, Kyle Hippe, Ryan Lewis, Aileen Cleary, Doga Ozgulbas, Rory Butler, Casey Stone, Abraham Stroka, Rafael Vescovi, and Ian Foster. Exploring benchmarks for self-driving labs using color matching. In *Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*, pp. 2147–2152, 2023.

Kam Hamidieh. A data-driven statistical model for predicting the critical temperature of a superconductor. *Computational Materials Science*, 154:346–354, 2018.

Nikolaus Hansen. Benchmarking a bi-population cma-es on the bbob-2009 function testbed. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, GECCO '09, pp. 2389–2396, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585055. doi: 10.1145/1570256.1570333. URL `https://doi.org/10.1145/1570256.1570333`.

Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 03 2003. ISSN 1063-6560. doi: 10.1162/106365603321828970. URL `https://doi.org/10.1162/106365603321828970`.

Nikolaus Hansen, Anne Auger, Raymond Ros, Olaf Mersmann, Tea Tušar, and Dimo Brockhoff and. Coco: a platform for comparing continuous optimizers in a black-box setting. *Optimization Methods and Software*, 36(1):114–144, 2021. doi: 10.1080/10556788.2020.1808977. URL `https://doi.org/10.1080/10556788.2020.1808977`.

Florian Häse, Loïc M Roch, and Alán Aspuru-Guzik. Next-generation experimentation with self-driving laboratories. *Trends in Chemistry*, 1(3):282–291, 2019.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors, 2012. URL `https://arxiv.org/abs/1207.0580`.

Yue Kang, Hang Yin, and Christian Berger. Test your self-driving algorithm: An overview of publicly available driving datasets and virtual testing environments. *IEEE Transactions on Intelligent Vehicles*, 4 (2):171–185, 2019.

Beomjoon Kim, Kyungjae Lee, Sungbin Lim, Leslie Kaelbling, and Tomas Lozano-Perez. Monte carlo tree search in continuous spaces using voronoi optimistic optimization with regret bounds. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:9916–9924, 4 2020a. doi: 10.1609/aaai.v34i06.6546. URL `https://ojs.aaai.org/index.php/AAAI/article/view/6546`.

Beomjoon Kim, Kyungjae Lee, Sungbin Lim, Leslie Kaelbling, and Tomas Lozano-Perez. Monte carlo tree search in continuous spaces using voronoi optimistic optimization with regret bounds. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(06):9916–9924, 2020b. doi: 10.1609/aaai.v34i06.6546. URL `https://ojs.aaai.org/index.php/AAAI/article/view/6546`.

Takatsugu Kosugi and Masahito Ohue. Design of cyclic peptides targeting protein–protein interactions using alphafold. *International Journal of Molecular Sciences*, 24, 9 2023. ISSN 14220067. doi: 10.3390/ijms241713257.

Andrew Leaver-Fay, Michael Tyka, Steven M Lewis, Oliver F Lange, James Thompson, Ron Jacak, Kristian W Kaufmann, P Douglas Renfrew, Colin A Smith, Will Sheffler, Ian W Davis, Seth Cooper, Adrien Treuille, Daniel J Mandell, Florian Richter, Yih-En Andrew Ban, Sarel J Fleishman, Jacob E Corn, David E Kim, Sergey Lyskov, Monica Berrondo, Stuart Mentzer, kk Zoran Popovic, James J Havranek, John Karanicolas, Rhiju Das, Jens Meiler, Tanja Kortemme, Jeffrey J Gray, Brian Kuhlman, David Baker, and Philip Bradley. Rosetta3: An object-oriented software suite for the simulation and design of macromolecules. *Methods in Enzymology*, 2011. doi: 10.1016/S0076-6879(11)87019-9.

Benjamin Letham, Roberto Calandra, Akshara Rai, and Eytan Bakshy. Re-examining linear embeddings for high-dimensional bayesian optimization, 2020. URL `https://arxiv.org/abs/2001.11659`.

Junqi Li, Steven G Ballmer, Eric P Gillis, Seiko Fujii, Michael J Schmidt, Andrea ME Palazzolo, Jonathan W Lehmann, Greg F Morehouse, and Martin D Burke. Synthesis of many different types of organic small molecules using one automated process. *Science*, 347(6227):1221–1226, 2015.

Lu Li, Qinqin Wang, Fanfan Wu, Qiaoling Xu, Jinpeng Tian, Zhiheng Huang, Qinghe Wang, Xuan Zhao, Qinghua Zhang, Qinkai Fan, Xiuzhen Li, Yalin Peng, Yangkun Zhang, Kunshan Ji, Aomiao Zhi, Huacong Sun, Mingtong Zhu, Jundong Zhu, Nianpeng Lu, Ying Lu, Shuopei Wang, Xuedong Bai, Yang Xu, Wei Yang, Na Li, Dongxia Shi, Lede Xian, Kaihui Liu, Luojun Du, and Guangyu Zhang. Epitaxy of wafer-scale single-crystal mos2 monolayer via buffer layer control. *Nature Communications*, 15(1):1825, 2024.

Jialin Liu, Antoine Moreau, Mike Preuss, Baptiste Roziere, Jeremy Rapin, Fabien Teytaud, and Olivier Teytaud. Versatile black-box optimization, 2020.

Alfred Ludwig. Discovery of new materials using combinatorial synthesis and high-throughput characterization of thin-film materials libraries combined with computational methods. *npj Computational Materials*, 5(1): 70, 2019.

J Madsen and T Susi. The abtem code: transmission electron microscopy from first principles [version 2; peer review: 2 approved]. *Open Research Europe*, 1(24), 2021. doi: 10.12688/openreseurope.13015.2.

J Sebastián Manzano, Wenduan Hou, Sergey S Zalesskiy, Przemyslaw Frei, Hsin Wang, Philip J Kitson, and Leroy Cronin. An autonomous portable platform for universal chemical synthesis. *Nature Chemistry*, 14 (11):1311–1318, 2022.

Hector G Martin, Tijana Radivojevic, Jeremy Zucker, Kristofer Bouchard, Jess Sustarich, Sean Peisert, Dan Arnold, Nathan Hillson, Gyorgy Babnigg, Jose M Marti, et al. Perspectives for self-driving labs in synthetic biology. *Current Opinion in Biotechnology*, 79:102881, 2023.

Amil Merchant, Simon Batzner, Samuel S Schoenholz, Muratahan Aykol, Gowoon Cheon, and Ekin Dogus Cubuk. Scaling deep learning for materials discovery. *Nature*, 624(7990):80–85, 2023.

Olaf Mersmann, Bernd Bischl, Heike Trautmann, Mike Preuss, Claus Weihs, and Günter Rudolph. Exploratory landscape analysis. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, GECCO '11, pp. 829–836, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450305570. doi: 10.1145/2001576.2001690. URL `https://doi.org/10.1145/2001576.2001690`.

Rémi Munos. Optimistic optimization of a deterministic function without the knowledge of its smoothness. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL `https://proceedings.neurips.cc/paper_files/paper/2011/file/7e889fb76e0e07c11733550f2a6c7a5a-Paper.pdf`.

Edin Muratspahić, Kristine Deibler, Jianming Han, Nataša Tomašević, Kirtikumar B Jadhav, Aina-Leonor Olivé-Marti, Nadine Hochrainer, Roland Hellinger, Johannes Koehbach, Jonathan F Fay, Mohammad Homaidur Rahman, Lamees Hegazy, Timothy W Craven, Balazs R Varga, Gaurav Bhardwaj, Kevin Appourchaux, Susruta Majumdar, Markus Muttenthaler, Parisa Hosseinzadeh, David J Craik, Mariana Spetea, Tao Che, David Baker, and Christian W Gruber. Design and structural validation of peptide–drug conjugate ligands of the kappa-opioid receptor. *Nature Communications*, 14:8064, 2023. ISSN 2041-1723. doi: 10.1038/s41467-023-43718-w. URL `https://doi.org/10.1038/s41467-023-43718-w`.

Masahiro Nomura and Masashi Shibata. cmaes : A simple yet practical python library for cma-es, 2024. URL `https://arxiv.org/abs/2402.01373`.

Paula C. Ortet, Samantha N. Muellers, Lauren A. Viarengo-Baker, Kristina Streu, Blair R. Szymczyna, Aaron B. Beeler, Karen N. Allen, and Adrian Whitty. Recapitulating the binding affinity of nrf2 for keap1 in a cyclic heptapeptide, guided by nmr, x-ray crystallography, and machine learning. *Journal of the American Chemical Society*, 143(10):3779–3793, 2021. doi: 10.1021/jacs.0c09799. URL `https://doi.org/10.1021/jacs.0c09799`. PMID: 33683866.

Martin Pincus. Letter to the Editor—A Monte Carlo Method for the Approximate Solution of Certain Types of Constrained Optimization Problems. *Operations Research*, 18(6):1225–1228, December 1970. doi: 10.1287/opre.18.6.1225. URL `https://ideas.repec.org/a/inm/oropre/v18y1970i6p1225-1228.html`.

Ziyuan Rao, Po-Yen Tung, Ruiwen Xie, Ye Wei, Hongbin Zhang, Alberto Ferrari, TPC Klaver, Fritz Körmann, Prithiv Thoudden Sukumar, Alisson Kwiatkowski da Silva, et al. Machine learning–enabled high-entropy alloy discovery. *Science*, 378(6615):78–85, 2022.

J. Rapin and O. Teytaud. Nevergrad - A gradient-free optimization platform. `https://GitHub.com/FacebookResearch/Nevergrad`, 2018.

Jacob T Rapp, Bennett J Bremer, and Philip A Romero. Self-driving laboratories to autonomously navigate the protein fitness landscape. *Nature Chemical Engineering*, 1(1):97–107, 2024.

Marcus Reis, Filipp Gusev, Nicholas G Taylor, Sang Hun Chung, Matthew D Verber, Yueh Z Lee, Olexandr Isayev, and Frank A Leibfarth. Machine-learning-guided discovery of 19f mri agents enabled by automated copolymer synthesis. *Journal of the American Chemical Society*, 143(42):17677–17689, 2021.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B. Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM Comput. Surv.*, 54(9), October 2021. ISSN 0360-0300. doi: 10.1145/3472291. URL `https://doi.org/10.1145/3472291`.

Blake T. Riley, Olga Ilyichova, Simon J. de Veer, Joakim E. Swedberg, Emily Wilson, David E. Hoke, Jonathan M. Harris, and Ashley M. Buckle. Klk4 inhibition by cyclic and acyclic peptides: Structural and dynamical insights into standard-mechanism protease inhibitors. *Biochemistry*, 58(21):2524–2533, 2019. doi: 10.1021/acs.biochem.9b00191. URL `https://doi.org/10.1021/acs.biochem.9b00191`. PMID: 31058493.

Semion K Saikin, Christoph Kreisbeck, Dennis Sheberla, Jill S Becker, and Alán Aspuru-Guzik. Closed-loop discovery platform integration is needed for artificial intelligence to make an impact in drug discovery. *Expert opinion on drug discovery*, 14(1):1–4, 2019.

Benjamin H Savitzky, Steven E Zeltmann, Lauren A Hughes, Hamish G Brown, Shiteng Zhao, Philipp M Pelz, Thomas C Pekin, Edward S Barnard, Jennifer Donohue, Luis Rangel DaCosta, Ellis Kennedy, Yujun Xie, Matthew T Janish, Matthew M Schneider, Patrick Herring, Chirranjeevi Gopal, Abraham Anapolsky, Rohan Dhall, Karen C Bustillo, Peter Ercius, Mary C Scott, Jim Ciston, Andrew M Minor, and Colin Ophus. py4DSTEM: A Software Package for Four-Dimensional Scanning Transmission Electron Microscopy Data Analysis. *Microscopy and Microanalysis*, 27(4):712–743, 08 2021. ISSN 1431-9276. doi: 10.1017/S1431927621000477. URL `https://doi.org/10.1017/S1431927621000477`.

Martin Seifrid, Robert Pollice, Andres Aguilar-Granda, Zamyla Morgan Chan, Kazuhiro Hotta, Cher Tian Ser, Jenya Vestfrid, Tony C Wu, and Alan Aspuru-Guzik. Autonomous chemical experiments: Challenges and perspectives on establishing a self-driving lab. *Accounts of Chemical Research*, 55(17):2454–2466, 2022.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. URL `http://www.ibm.com/software/commerce/optimization/cplex-optimizer/`.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016. doi: 10.1109/JPROC.2015.2494218.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 2016. ISSN 1476-4687. doi: 10.1038/nature16961. URL `https://doi.org/10.1038/nature16961`.

Jiamei Song, Christopher S Allen, Si Gao, Chen Huang, Hidetaka Sawada, Xiaoqing Pan, Jamie Warner, Peng Wang, and Angus I Kirkland. Atomic resolution defocused electron ptychography at low dose with a fast, direct electron detector. *Scientific Reports*, 9:3919, 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-40413-z. URL `https://doi.org/10.1038/s41598-019-40413-z`.

Jost Tobias Springenberg, Aaron Klein, Stefan Falkner, and Frank Hutter. Bayesian optimization with robust bayesian neural networks. *Advances in Neural Information Processing Systems*, 29, 2016. URL `https://proceedings.neurips.cc/paper_files/paper/2016/file/a96d3afec184766bfeca7a9f989fc7e7-Paper.pdf`.

Rainer Storn and Kenneth Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997. ISSN 1573-2916. doi: 10.1023/A:1008202821328. URL `https://doi.org/10.1023/A:1008202821328`.

Nathan J Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E Kumar, Tanjin He, David Milsted, Matthew J McDermott, Max Gallant, Ekin Dogus Cubuk, Amil Merchant, et al. An autonomous laboratory for the accelerated synthesis of novel materials. *Nature*, 624(7990):86–91, 2023.

Daniel P Tabor, Loïc M Roch, Semion K Saikin, Christoph Kreisbeck, Dennis Sheberla, Joseph H Montoya, Shyam Dwaraknath, Muratahan Aykol, Carlos Ortiz, Hermann Tribukait, et al. Accelerating the discovery of materials for clean energy in the era of smart automation. *Nature reviews materials*, 3(5):5–20, 2018.

Jose Antonio Garrido Torres, Sii Hong Lau, Pranay Anchuri, Jason M Stevens, Jose E Tabora, Jun Li, Alina Borovika, Ryan P Adams, and Abigail G Doyle. A multi-objective active learning platform and web app for reaction optimization. *Journal of the American Chemical Society*, 144(43):19999–20007, 2022.

Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-bench: Benchmarks for data-driven offline model-based optimization. In *International Conference on Machine Learning*, pp. 21658–21676. PMLR, 2022.

Harold Varmus, R Klausner, E Zerhouni, T Acharya, AS Daar, and PA Singer. Grand challenges in global health, 2003.

Alexander A. Vinogradov, Yizhen Yin, and Hiroaki Suga. Macrocyclic peptides as drug candidates: Recent progress and remaining challenges. *Journal of the American Chemical Society*, 141:4167–4181, 3 2019. ISSN 0002-7863. doi: 10.1021/jacs.8b13178. URL `https://doi.org/10.1021/jacs.8b13178`. doi: 10.1021/jacs.8b13178.

Amanda A Volk, Robert W Epps, Daniel T Yonemoto, Benjamin S Masters, Felix N Castellano, Kristofer G Reyes, and Milad Abolhasani. Alphaflow: autonomous discovery and optimization of multi-step chemistry using a self-driven fluidic lab guided by reinforcement learning. *Nature Communications*, 14(1):1403, 2023.

Linnan Wang, Rodrigo Fonseca, and Yuandong Tian. Learning search space partition for black-box optimization using monte carlo tree search. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Colin White, Willie Neiswanger, and Yash Savani. Bananas: Bayesian optimization with neural architectures for neural architecture search. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12): 10293–10301, May 2021. doi: 10.1609/aaai.v35i12.17233. URL `https://ojs.aaai.org/index.php/AAAI/article/view/17233`.

Chris Ying, Aaron Klein, Eric Christiansen, Esteban Real, Kevin Murphy, and Frank Hutter. NAS-bench-101: Towards reproducible neural architecture search. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7105–7114, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL `http://proceedings.mlr.press/v97/ying19a.html`.

Alessandro Zorzi, Kaycie Deyle, and Christian Heinis. Cyclic peptide therapeutics: past, present and future, 2017.

A Appendix

A.1 Synthetic functions

The synthetic functions are designed to evaluate and analyze computational optimization approaches. In total, eig of them are selected based on their physical properties and topologies. The Ackley function can be written as:

$$f(x) = -a \cdot exp(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2} - exp(\frac{1}{d}\sum_{i=1}^{d} cos(cx_i)) + a + exp(1), \tag{5}$$

where $a = 20$, $b = 0.2$, $c = 2\pi$, and d is the dimension.

The Rosenbrock function can be written as:

$$f(x) = \sum_{i=1}^{d-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]. \tag{6}$$

The Rastrigin function can be written as:

$$f(x) = 10d + \sum_{i=1}^{d-1}[x_i^2 - 10\cos(2\pi x_i)]. \tag{7}$$

The three functions are evaluated on the hypercube $x_i \in [-5, 5]$, for all $i = 1, \dots, d$ with a discrete search space of a step size of 0.1.

The Schwefel function can be written as:

$$f(x) = 418.9828d - \sum_{i=1}^{d} x_i \sin(\sqrt{|x_i|}), \tag{8}$$

where d is the dimension. The function is evaluated on the hypercube $x_i \in [-500, 500]$, for all $i = 1, \dots, d$ with a discrete search space of a step size of 1.

The Griewank function can be written as:

$$f(x) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos(\frac{x_i}{\sqrt{i}}) + 1, \tag{9}$$

where d is the dimension. The function is evaluated on the hypercube $x_i \in [-600, 600]$, for all $i = 1, \dots, d$ with a discrete search space of a step size of 1.

The Michalewicz function can be written as:

$$f(x) = -\sum_{i=1}^{d} \sin(x_i)\sin^{2m}(\frac{ix_i^2}{\pi}), \tag{10}$$

where d is the dimension. The function is evaluated on the hypercube $x_i \in [0, \pi]$, for all $i = 1, \dots, d$ with a discrete search space of a step size of 10^{-4}.

The Levy function can be written as:

$$f(x) = \sin^2(\pi\omega_1) + \sum_{i=1}^{d-1}(\omega_i - 1)^2[1 + 10\sin^2(\pi\omega_i + 1)] + (\omega_d - 1)^2[1 + \sin^2(2\pi\omega_d)], \tag{11}$$

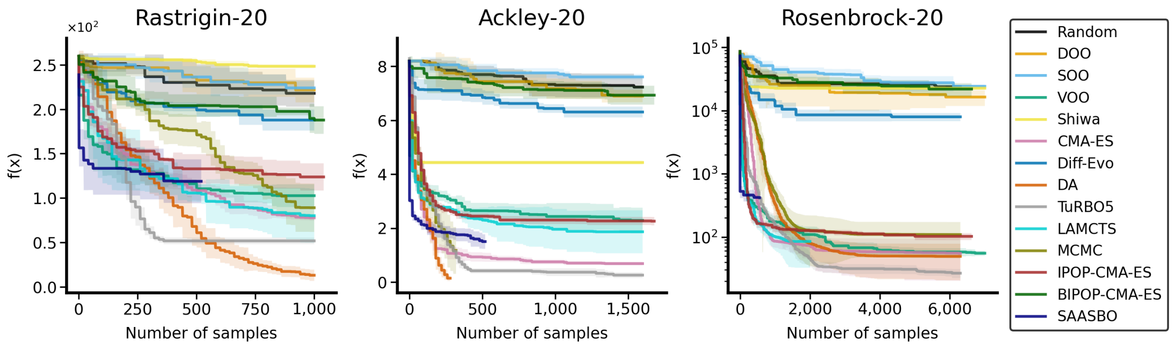

Figure A1: Evaluation of sampling efficiency for Ackley, Rastrigin, Rosenbrock in 20-dimension. No single method demonstrates consistent superiority across all scenarios.

where $\omega_i = 1 + \frac{x_i - 1}{4}$, and d is the dimension. The function is evaluated on the hypercube $x_i \in [-10, 10]$, for all $i = 1, \ldots, d$ with a discrete search space of a step size of 0.1.

The Sphere function can be written as:

$$f(x) = \sum_{i=1}^{d} x_i^2, \tag{12}$$

where d is the dimension. The function is evaluated on the hypercube $x_i \in [-5, 5]$, for all $i = 1, \ldots, d$ with a discrete search space of a step size of 0.1.

Relative Gaussian noise is introduced to the synthetic functions to better emulate real-world experimental uncertainty. The noise level is controlled by the standard deviation of the Gaussian distribution, which scales with the function value as follows:

$$\sigma = |y| \times \text{noise\_level} \tag{13}$$

where *noise_level* denotes the relative magnitude of the perturbation.

A.2 Data sample efficiency

Figure A1 shows the history of the active optimization performance to evaluate the sampling efficiency of the algorithms with 20 dimensions. Similar to Figure 5, 14 methods are evaluated against the current minimum across different data acquisition scenarios. Consistent with obversation in high dimensional problems, no single method demonstrates consistent dominance across all tasks. For the Ackley-20 function, DA and MCMC demonstrate rapid convergence to the global minimum of $f(x)$. In the Rastrigin-20 function, TuRBO5 and DA outperform other approaches. Interestingly, search methods such as TuRBO5 constantly achieves lower values, whereas others, e.g., Diff-Evo, appear to become trapped in local minima.

A.3 Noisy synthetic functions

Table A1, Figure A2, and Figure A3 present the benchmark results, optimization histories, and standard deviations for the selected functions and methods under varying levels of Gaussian noise. The results indicate that adding noise exerts a complex influence on the optimization process; In general, algorithmic performance tends to degrade, and variability increases as the noise level rises. Notably, TuRBO consistently achieves the best performance across nearly all functions and different noise levels.

A.4 Surrogate model setups

Training details We used 1D convolutional neural networks (1D-CNN) as the surrogate model to fit the synthetic functions. We initiated each surrogate model training with 2,000 uniformly sampled data points from the parameter space of the corresponding synthetic function to train the surrogate model, where 1,600

Table A1: Evaluations of AO search strategies on synthetic functions under varying Gaussian noise, where the values with bold texts denote the best optimization result across all the methods. All the functions are evaluated in 20 dimensions. Results are averaged over 5 trials, and $\pm$ denotes the standard deviation.

	Ackley 0% noise	**Ackley 5% noise**	**Ackley 10% noise**	**Ackley 20% noise**	**Rosenbrock 0% noise** ($\times 10^4$)	**Rosenbrock 5% noise** ($\times 10^4$)	**Rosenbrock 10% noise** ($\times 10^4$)	**Rosenbrock 20% noise** ($\times 10^4$)	**Schwefel 0% noise** ($\times 10^3$)	**Schwefel 5% noise** ($\times 10^3$)	**Schwefel 10% noise** ($\times 10^3$)	**Schwefel 20% noise** ($\times 10^3$)
$f(\mathbf{x}^*)$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-19.63	0	0	0	0
TuRBO5	0.37 ± 0.14	$\mathbf{0.91 \pm 0.17}$	$\mathbf{0.86 \pm 0.28}$	$\mathbf{0.93 \pm 0.28}$	$\mathbf{0.003 \pm 0.001}$	0.008 ± 0.002	0.009 ± 0.004	0.008 ± 0.003	$\mathbf{2.88 \pm 0.31}$	$\mathbf{2.76 \pm 0.45}$	$\mathbf{3.10 \pm 0.66}$	$\mathbf{3.32 \pm 0.48}$
DA	$\mathbf{0.00 \pm 0.00}$	6.21 ± 0.27	5.99 ± 0.37	6.11 ± 0.67	0.004 ± 0.003	0.219 ± 0.068	0.310 ± 0.116	0.307 ± 0.144	3.16 ± 0.28	3.01 ± 0.63	3.24 ± 0.26	3.52 ± 0.25
MCMC	0.005 ± 0.001	2.36 ± 0.58	2.16 ± 1.06	2.01 ± 0.73	0.021 ± 0.003	$\mathbf{0.002 \pm 0.002}$	$\mathbf{0.002 \pm 0.002}$	$\mathbf{0.002 \pm 0.002}$	4.94 ± 0.34	4.57 ± 0.78	4.78 ± 0.36	5.10 ± 0.42

All benchmark tasks here involve minimization objectives.

The asterisk (*) represents the global minimum of the function.

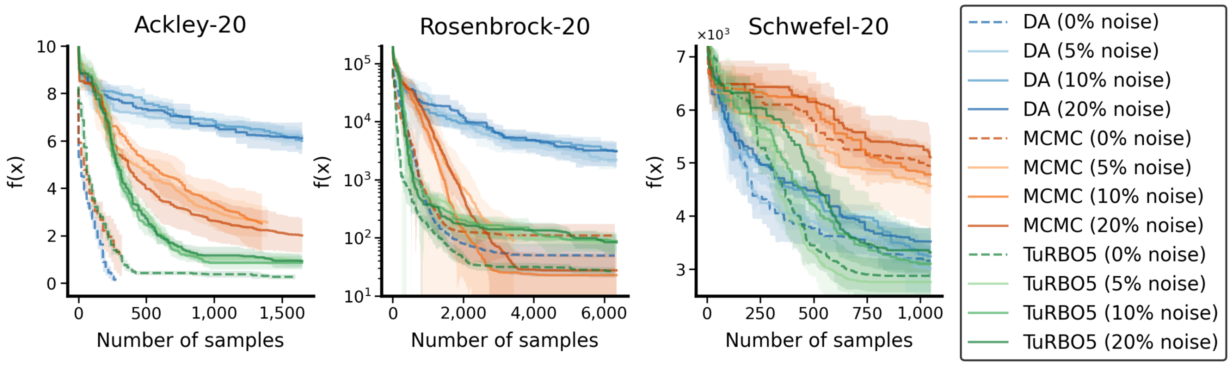

Figure A2: Comparison of DA, MCMC, and TuRBO5 on 20-dimension Ackley, Rosenbrock, and Schwefel functions under varying Gaussian noise to $f(x)$. The shaded regions denotes standard deviation over 5 runs.

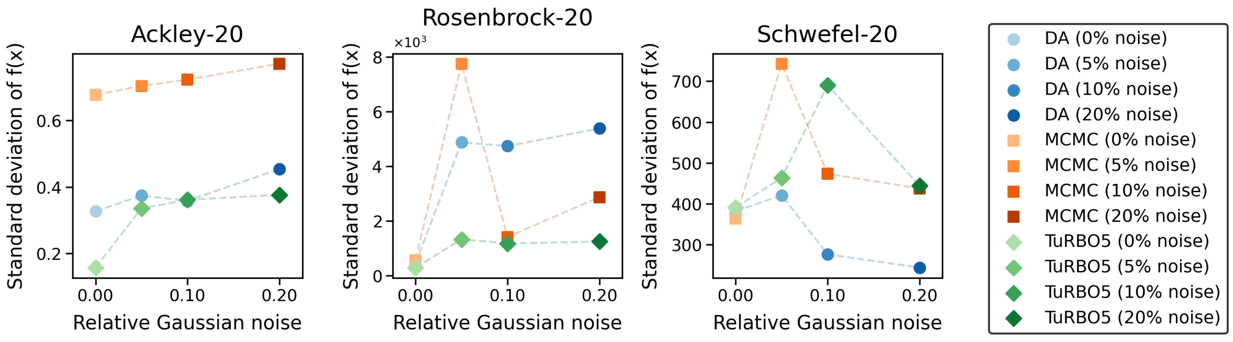

Figure A3: Average standard deviation of $f(x)$ over the optimization history for DA, MCMC, and TuRBO5 on 20-dimension Ackley, Rosenbrock, and Schwefel functions under increasing relative Gaussian noise to f(x).

samples were used for the training set and 400 samples for the testing set. Adam Optimizer was employed with a learning rate of 0.001, and the activation function utilized is the Exponential Linear Unit (ELU). The loss function is the mean square error (MSE) for all synthetic functions except Rastrigin where we used mean absolute percentage error (MAPE). The 1D-CNN model is trained for 500 epochs with early stopping patience of 30 and a batch size of 64. Additionally, the outputs for some of the functions are transformed to avoid the scaling problem for surrogate model training, the corresponding transformation (if applied) is defined in the corresponding sections as follows.

Ackley The 1D-CNN comprises 5 convolutional layers with filter sizes of 128, 64, 32, 16, and 8 respectively, each using a kernel size of 3. It also includes 2 max-pooling layers with a pooling size of 2, 2 dropout layers with a dropout rate of 0.2, followed by a flatten layer, 2 fully connected layers with 128 and 64 units

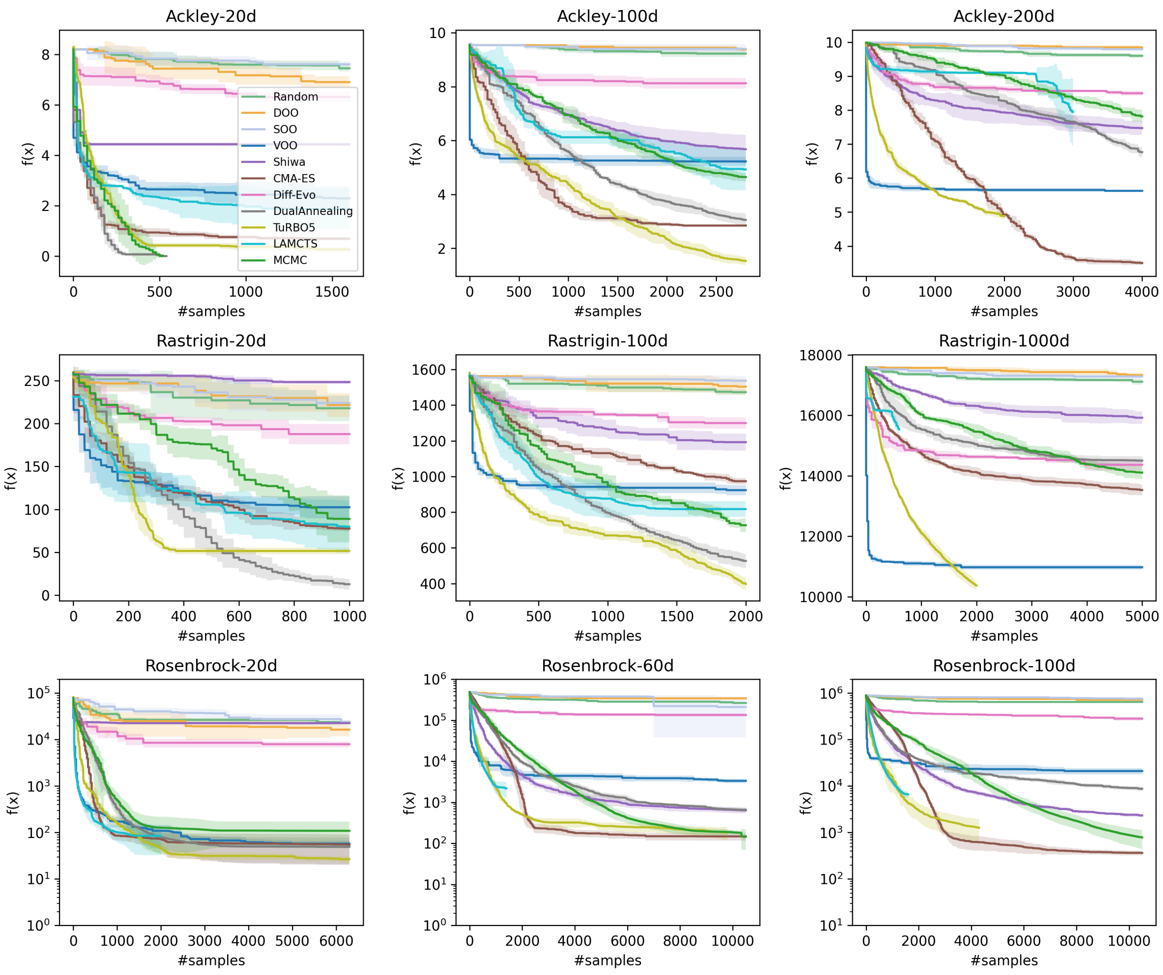

Figure A4: Additional experiments on the influence of dimensionality on the performance.

respectively, and an output layer. To obtain a better fitting, we employed a transformation of $100/(f(\mathbf{x})+0.01)$ to the output of the Ackley function $f(\mathbf{x})$.

Rastrigin The 1D-CNN consists of 6 convolutional layers with filter sizes of 256, 128, 64, 32, 16, and 8 respectively. The kernel sizes are 5, 5, 3, 3, 3, and 3 respectively, with strides of 1, 2, 2, 1, 1, and 1 respectively. Following these convolutional layers is a flatten layer, 2 fully connected layers with 128 and 64 units respectively, and an output layer.

Rosenbrock The 1D-CNN comprises 6 convolutional layers with filter sizes of 128, 64, 32, 16, 8, and 4 respectively, each using a kernel size of 3. Additionally, there are 3 max-pooling layers with a pooling size of 2, 2 dropout layers with a dropout rate of 0.2, followed by a flatten layer, 1 fully connected layer with 64 units, and an output layer. To obtain a better fitting, we employed a transformation of $100/(f(\mathbf{x})/100d + 0.01)$ to the output of the Rosenbrock function $f(\mathbf{x})$ in its d-dimensional form.

Griewank The model architecture is the same as Rosenbrock. We employed the transformation $10/(f(\mathbf{x})/d+0.001)$ to the output of the Griewank function $f(\mathbf{x})$ in its d-dimensional form.

Schwefel The 1D-CNN consists of 7 convolutional layers with filter sizes of 256, 128, 64, 32, 16, 8, and 4 respectively. The kernel size is set to 5 with a stride of 1 for all layers. These are followed by a flatten layer, 6 fully connected layers with 128, 64, 32, 16, and 8, respectively, and an output layer. We re-scaled the output of the Schwefel function $f(\mathbf{x})$ with a factor of 0.01.

Michalewicz The 1D-CNN comprises 5 convolutional layers with filter sizes of 128, 64, 32, 16, and 8 respectively, each using a kernel size of 3 with a stride of 1. Additionally, there are 3 max-pooling layers with a pooling size of 2, 2 dropout layers with a dropout rate of 0.2, followed by a flatten layer, 1 fully connected layer with 64 units, and an output layer.

Levy The 1D-CNN comprises 6 convolutional layers with filter sizes of 128, 64, 32, 16, 8, and 4 respectively, each using a kernel size of 3 with a stride of 1. Additionally, there are 3 max-pooling layers with a pooling size of 2, 2 dropout layers with a dropout rate of 0.2, followed by a flatten layer, 1 fully connected layer with 64 units, and an output layer.

Sphere The model architecture is the same as Levy.

A.5 Setups for AO search strategies

For the benchmark of synthetic function tasks, the AO search strategies were conducted without information on the ground truth oracle functions. The implementations of VOO, SOO, and DOO were sourced from an established repository [1], while the methods including CMA-ES, Differential Evolution (Diff-Evo), and Dual Annealing (DA) were derived from the Scipy optimize module, and Shiwa was obtained from Nevergrad [2]. The implementation of Bayesian Optimization is from [3]. The implementation of TuRBO5 is from [4]. The implementation of LAMCTS is from [5]. The implementation of IPOP-CMA-ES and BIPOP-CMA-ES is from [6]. The implementation of SAASBO is from [7](Balandat et al., 2020). All algorithms were employed with the default setting in the reference implementation.

A.6 Additional Neural network architecture search details

NAS is an automated approach for identifying optimal neural network architectures by systematically exploring and evaluating a wide range of network configurations to achieve superior performance on a specific task.

Dataset and optimization target To benchmark the efficacy of AO search strategies in optimizing neural network structures within the context of active learning, we choose the NAS-Bench-101 dataset (Ying et al., 2019), which contains over 400,000 unique convolutional neural networks along with their corresponding performance metrics, trained on the CIFAR-10 dataset (Hinton et al., 2012). Each neural network is represented by a 7×7 upper-triangular adjacency matrix with up to 9 edges, where nodes represent specific operations and edges denote the connection relationships between these operations. The first operation represents the input, and the last represents the output, while the remaining five components can be selected from 3×3 convolution, 1×1 convolution, or 3×3 max-pooling. The objective of the NAS task is to identify an optimized neural network structure that achieves the highest classification accuracy on the test set (test acc).

Neural network architecture encoding We adopt a truncated 40-bit path-based encoding scheme (White et al., 2021) to represent the neural network structure, where each bit corresponds to a specific path from the input layer to the output layer, incorporating various operators along the way. For optimization algorithms like CMA-ES, Dual Annealing, LAMCTS, and TuRBO5, which require a well-defined search domain, we parameterize the neural network structure into a 36-dimensional vector within the continuous [0, 1] space, as adopted from prior work (Letham et al., 2020). The first 21 entries correspond to the adjacency matrix, where the largest values set the respective elements in the matrix to 1. The remaining 15 entries

[1]`https://github.com/beomjoonkim/voot`
[2]`https://github.com/facebookresearch/nevergrad`
[3]`https://github.com/bayesian-optimization/BayesianOptimization`
[4]`https://github.com/uber-research/TuRBO`
[5]`https://github.com/facebookresearch/LaMCTS`
[6]`https://github.com/CyberAgentAILab/cmaes`
[7]`https://github.com/pytorch/botorch/blob/main/tutorials/saasbo/saasbo.ipynb`

represent the one-hot encoding of 5 components, each with three possible operations. For MCMC and Random Search, optimization is performed directly at the adjacency matrix level.

Surrogate model We train a 1D-CNN model to map the path encoding into the test acc. The 1D-CNN consists of 5 convolutional layers with filter sizes of 128, 64, 32, 16, and 8, respectively, each using a kernel size of 3. It also includes 2 max-pooling layers with a pooling size of 2, 2 dropout layers with a dropout rate of 0.2, followed by a flatten layer, 2 fully connected layers with 128 and 64 units, respectively, and a final output layer. The loss function used is mean square error (MSE).

Setups for AO search strategies The optimization process begins by generating 200 random initial data points from NAS-Bench-101, which are used to train the initial surrogate model. In the active learning loop, optimization algorithms then sample 20 optimized successors by refining the surrogate model, expanding the dataset. The updated surrogate model is subsequently used in the next iteration of the loop, continually improving the optimization process.

- **MCMC**: The acceptance rate is defined as $exp(-\delta/T)$, where δ represents the difference between the proposal point and the current best point. If $\delta > 0$, indicating the proposal point is better than the current best, the proposal is accepted outright; otherwise, it is accepted with the calculated acceptance rate. The temperature parameter, T, decreases exponentially with each iteration, starting at an initial value of 0.01, with a half-life of 200 iterations.
- **CMA-ES**: 0.25 $sigma_0$, 300 maxfevals, with other parameters using default settings.
- **DA**: 5 maxiter, 300 maxfun, with other parameters using default settings.
- **LAMCTS**: 40 ninits, 0.1 Cp, 100 iterations, with other parameters using default settings.
- **TuRBO5**: 50 n_init, 300 max_evals, 5 n_trust_regionsm, 10 batch_size, 2000 max_cholesky_size, 50 n_training_steps, with other parameters using default settings.

A.7 Additional Lunar landing problem details

The Lunar Lander problem is a widely recognized benchmark environment in the OpenAI Gym toolkit, frequently utilized in reinforcement learning research to evaluate control strategies. The task involves controlling a simulated lunar module to achieve a safe landing on the moon's surface.

Action Space and Reformulation The environment provides four discrete action options: (i) do nothing, (ii) fire the left engine, (iii) fire the main engine, and (iv) fire the right engine. While this problem is traditionally framed as a trajectory planning task with cumulative objectives, we reformulate it into a non-cumulative optimization problem by fixing the initial conditions. The goal is to design an optimal sequence of 100 discrete actions to maximize the reward, where the action space includes 0 (do nothing), 1 (fire left engine), 2 (fire main engine), and 3 (fire right engine). To ensure consistency, the environment reset seed is fixed at 42 to generate a consistent initial state.

Search algorithms The setups of the search algorithms in the AO pipelines are as follows:

- **Random**: Random seed is set to 42.
- **DOO**: 0.1 explr_p with other parameters using default settings.
- **SOO**: Default settings.
- **VOO**: 1 explr_p with other parameters using default settings.
- **Shiwa**: Default settings.
- **CMA-ES**: Default settings.

- **DA**: Default settings.
- **MCMC**: Default settings.

A.8 Additional cyclic peptide design details

Pipeline The pipeline for cyclic peptide SL consists of three components: (1) AlphaFold2 with cyclic offsets to predict the structure of protein-cyclic peptide complexes (Kosugi & Ohue, 2023); (2) ProteinMPNN to ensure the diversity of designed cyclic peptide sequences (Dauparas et al., 2022); and (3) Rosetta's interface analyzer to evaluate the quality of the designed interface (Leaver-Fay et al., 2011). Given the structure of the desired protein and the corresponding interaction hotspot, the pipeline begins with an optimization method that iteratively searches for the sequence yielding the highest AlphaFold2 pLDDT (predicted Local Distance Difference Test) score, which indicates the confidence level of the predicted structure. The optimized sequence is fed into ProteinMPNN to generate a pool of diverse sequences. Finally, the product of two Rosetta binding metrics—shape complementarity (SC) and the change in Solvent Accessible Surface Area ($dSASA$)—is used to filter the output sequences, with the best-fit design likely to have high SC and $dSASA$ values (Muratspahić et al., 2023).

Simulation Settings The structure of the protein and cyclic peptide complex is predicted using AlphaFold2-multimer with cyclic offsets, as implemented in ColabDesign (Kosugi & Ohue, 2023). ProteinMPNN is also employed in ColabDesign with a batch size of 128. The SC and $dSASA$ values for the predicted structure of the protein and cyclic peptide complex are computed using the PyRosetta Interface Analyzer (Chaudhury et al., 2010).

AO search strategy setups To ensure a fair comparison across AO search strategies, we limited the number of oracle function evaluations to approximately 1000. The specific settings are detailed as follows.

- **Diff-Evo**: a population size of 15 with a maximum of 1000 function evaluations.
- **DA**: 50 iterations with a maximum of 1000 function evaluations.
- **TuRBO5**: 20 initial samples with 5 trust regions, followed by up to a maximum of 1000 evaluations in batches of 5.
- **BO**: 50 initial samples followed by 950 iterations.

A.9 Additional electron ptychography details

Simulation settings The MoS_2 dataset uses an 80 kV probe energy, a 20 mrad probe-forming semi-angle, a set of probe aberration coefficients of defocus -130 Å, two-fold astigmatism (C12) 20 Å, two-fold astigmatism angle (Phi12) 0.785, three-fold astigmatism (C23) 15 Å, three-fold astigmatism angle (Phi23) 0.295, axial coma (C21) 30 Å, axial coma angle (Phi21) 0.534, spherical aberration (C30) -2×10^4 Å. The dataset consists of 51 diffraction patterns with a 0.312 Å scanning step size in the real space. In addition, all diffraction patterns in both datasets were corrupted with Poisson noise of 10,000 e/Å^2 for this task. The ptychographic reconstruction is performed with a multi-slice approach using py4DSTEM (Savitzky et al., 2021), a comprehensive open-source package for different modes of 4D-STEM data analysis.

Evaluation metrics In addition to the NMSE score, we evaluate the quality of electron ptychographic reconstruction using two extra metrics: probe and object reconstruction errors. First, the probe reconstruction error calculates the normalized mean square error between the reconstructed and the simulated probes in the real space. While the ptychographic algorithm itself does not have the access to the ground truth probe function, a successful ptychographic reconstruction must accurately retrieve both the probe function and the object transmission function. As we deliberately exaggerated the aberrations of the probe in the MoS_2 dataset, this metric can act as another useful metric to evaluate the reconstruction. Second, the object reconstruction error computes the normalized mean square error between the median-angle-annular-dark-field signal (without added noise) and the phase of the object transmission function. This metric directly demonstrates the quality of the retrieved object transmission function.

Table A2: Optimized reconstruction parameters by different AO search strategies for the MoS_2 dataset.

	semiangle cutoff (mrad)	**energy** (kV)	**number of iterations**	**step size**	**identical slices iteration**	**slice thicknesses** (Å)	**number of slices**	**defocus** (Å)	**C12** (Å)	**phi12** (rad)	**C30** (Å)	**C21** (Å)	**phi21** (rad)	**C23** (Å)	**phi23** (rad)
Ground truth	20.0	80	-	-	-	-	-	-130	20	0.79	-2.0×10^4	30	0.53	15	0.29
Diff-Evo	23.4	73	20	0.65	4	4.6	16	-185	6.0	0.95	-9.4×10^4	95	0.06	84	1.00
DA	22.0	269	18	0.87	33	5.4	21	-118	50.0	0.61	-4.1×10^4	62	0.15	47	0.87
TuRBO5	18.0	242	20	0.57	2	18.1	29	-8	4.0	0.60	-5.6×10^4	42	0.57	19	0.54
BO	22.4	254	10	0.71	2	34.4	17	-166	16.0	0.83	-5.4×10^4	89	0.3	16	0.03

Hyper-parameter settings We used 20 samples for initialization of all AO search strategies. We set the independent trust regions to 5 for TuRBO. The rest hyper-parameters take the default values for the individual AO search strategies.

Optimization results The optimization history (Figure A7) shows that TuRBO achieves the lowest NMSE after 500 samples, while other methods are trapped into local minima. Table A2 summarizes the reconstruction parameters for each AO search strategies. Figure A6 visualizes the reconstructed amplitude of the probe functions with different AO search strategies.

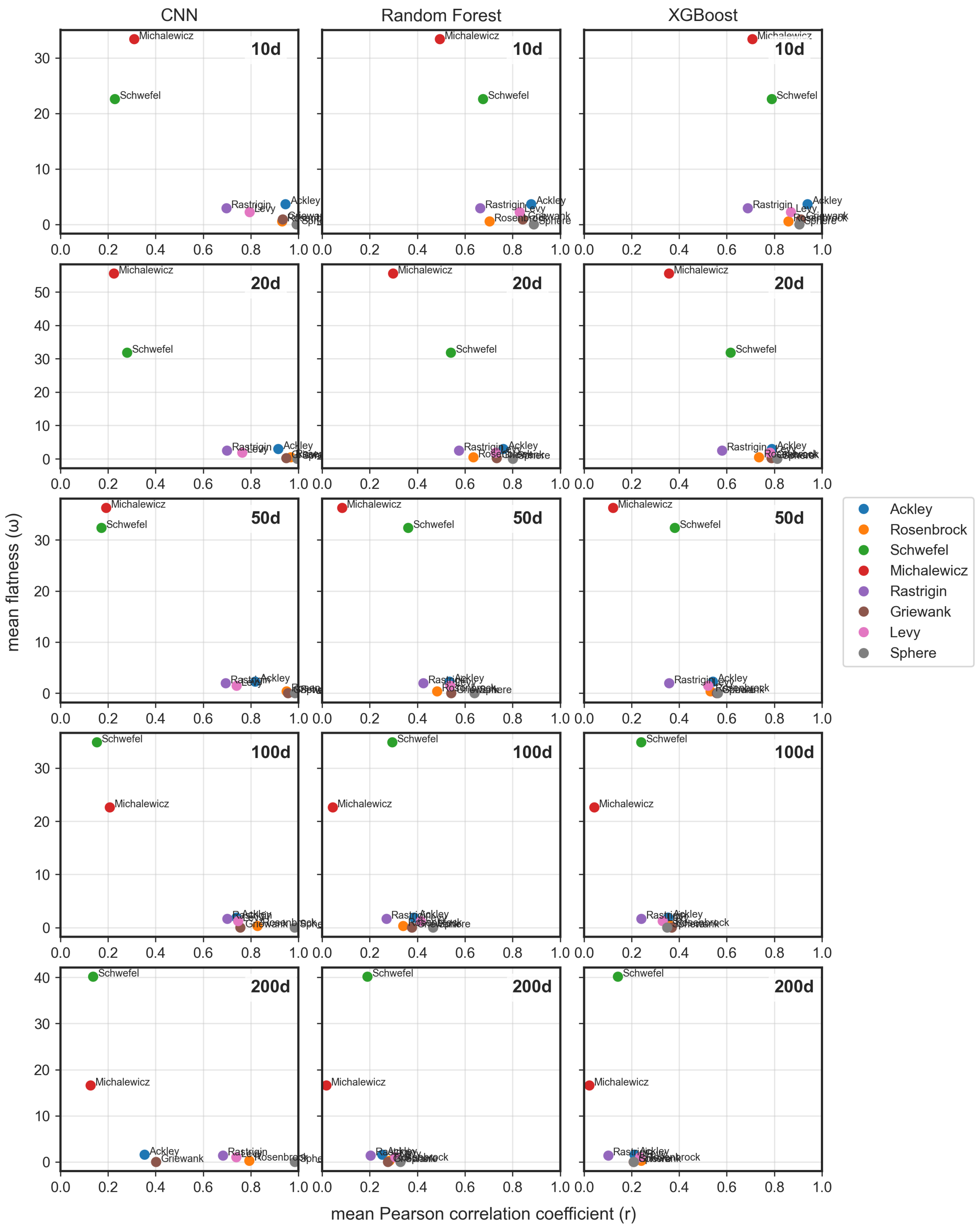

Figure A5: Mean Pearson correlation (r) vs. mean flatness (ω) for surrogate models (CNN, Random Forest, XGBoost) across benchmark functions and dimensions (10d–200d). Each point represents a function, color-coded by type.

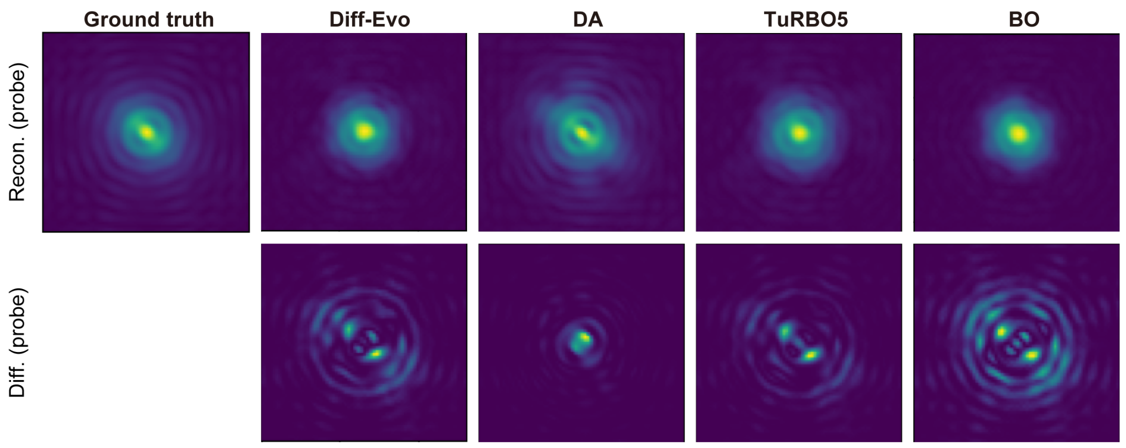

Figure A6: Visualization of amplitude (of the probe functions) reconstructed using parameters obtained from the corresponding AO search strategies. The second row visualizes the normalized mean square error between the ground truth and the reconstructed amplitude values.

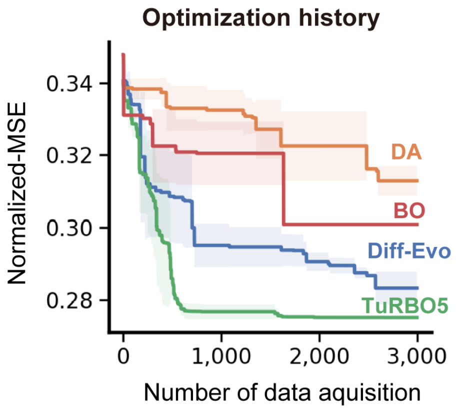

Figure A7: Optimization history of AO search strategies on the MoS_2 dataset.