

Tutorial:

Multi-Objective Bayesian Optimization

with BoTorch and Ax

2ND WORKSHOP ON AI FOR THE ELECTRON ION COLLIDER (AI4EIC)

OCTOBER 10, 2022



BoTorch



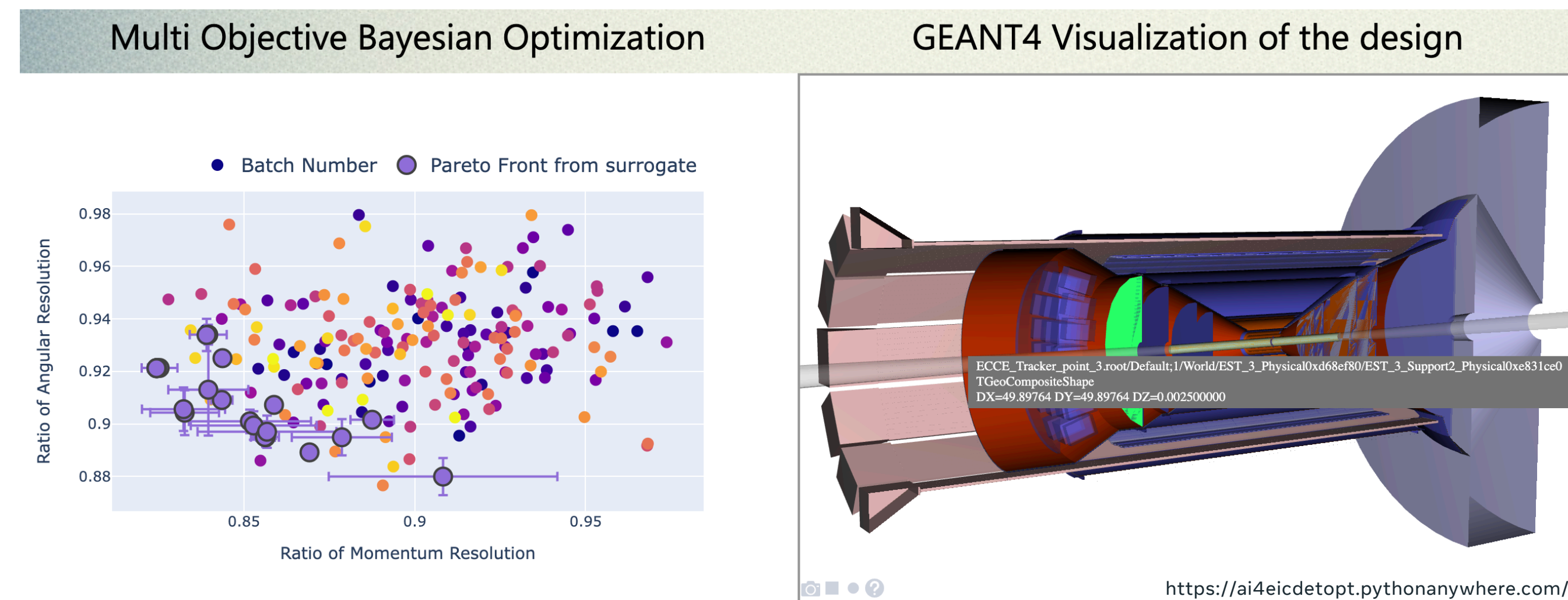
Ax

Maximilian Balandat

Adaptive Experimentation team, Meta

Motivation: ECCE Detector Design

- Domain: 11+ parameters (location of tracking layers, etc.)
- Multiple objectives:
 - resolution (momentum, polar angular)
 - efficiency
 - cost
 - ...
- Constraints on geometry (to avoid overlap)
- Evaluating objectives requires **computationally expensive** simulation for each design point (parameter configuration)

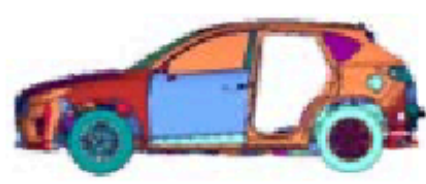
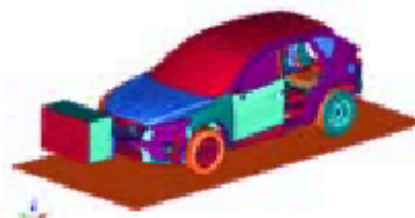



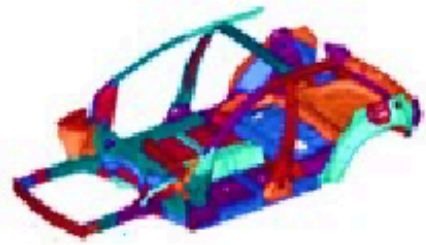
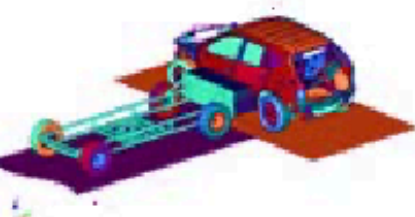
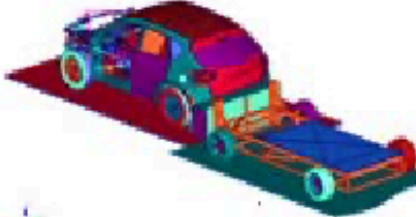
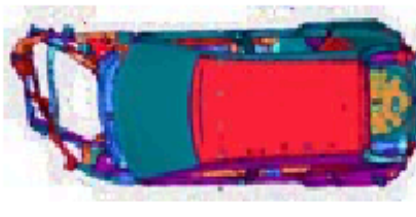



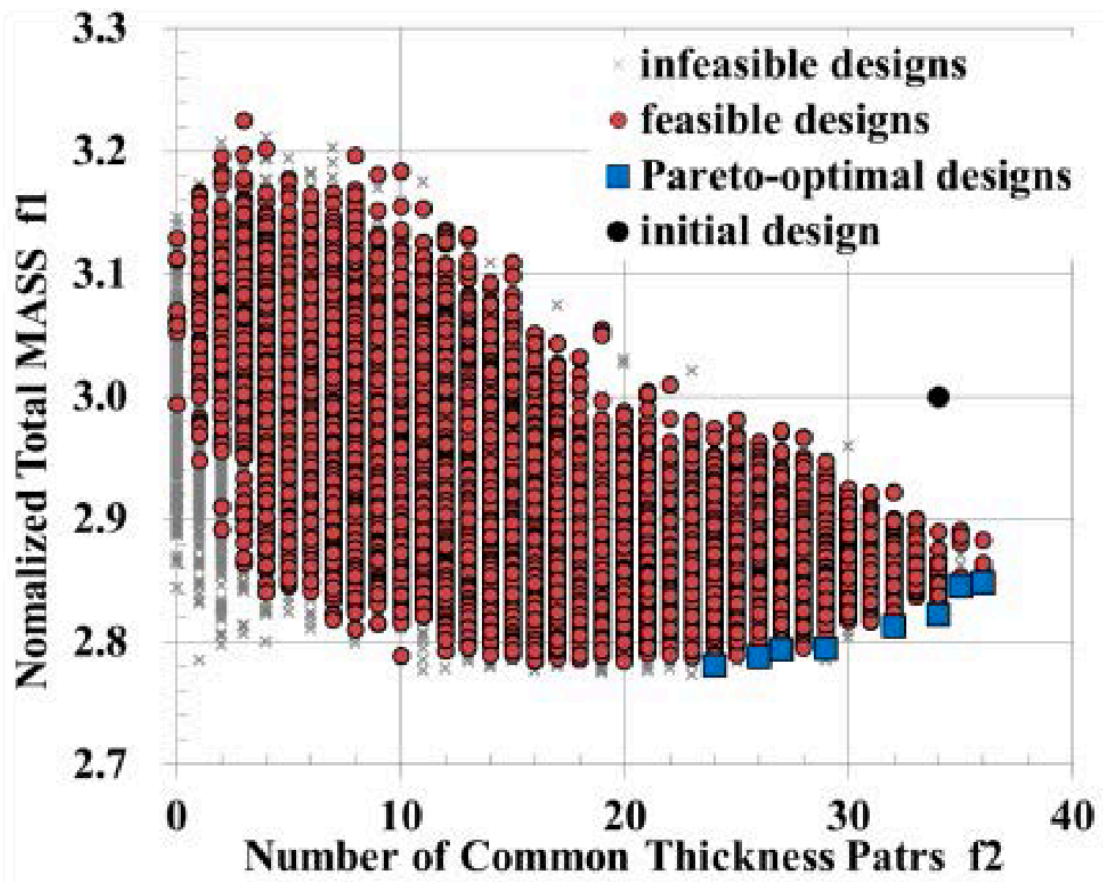
Goal: Explore a complex design space to optimally trade off multiple competing objectives that are computationally expensive-to-evaluate.

Motivation: Vehicle Design Optimization

- Domain: **222 parameters** (properties of structural parts)
- Goals:
 - **minimize** total vehicle mass of three vehicles (Mazda 3, 6, CX-5)
 - **maximize** number of parts shared across vehicles
 - satisfy **54 black-box outcome constraints** (performance & safety requirements)
- Evaluation requires solving a complex physics simulation
- Originally solved on the (at the time) world’s fastest super computer in **>3,000 CPU years**

➡ **56 outcomes**

Car Type	SUV	Constraint Conditions	Frontal 40% Offset Impact	Full Frontal Impact	Longitudinal Bending Mode	Torsion Mode	
							
Design Variables			Side Impact	Rear 70% Offset Impact	Lateral Bending	Torsion Stiffness	
							



Hot Take:

Every optimization problem is fundamentally
a multi-objective optimization problem.

Outline

1. Motivation & Intro
2. Background: Methodology
3. Applications
4. Tutorial: Hands-On Examples
5. Advanced Features

2 Background

2 BACKGROUND

Bayesian Optimization (BO)

- Goal: Sample-efficient optimization of “black-box” function: $\max_{x \in \mathcal{X}} f(x)$, where f is expensive to evaluate, w/o gradients
- Strategy: Leverage
 - a probabilistic **surrogate model** \hat{f}
 - an **acquisition function** $x \mapsto \alpha(\hat{f}(x))$ to guide exploration

Typical surrogate: Gaussian Process (GP)

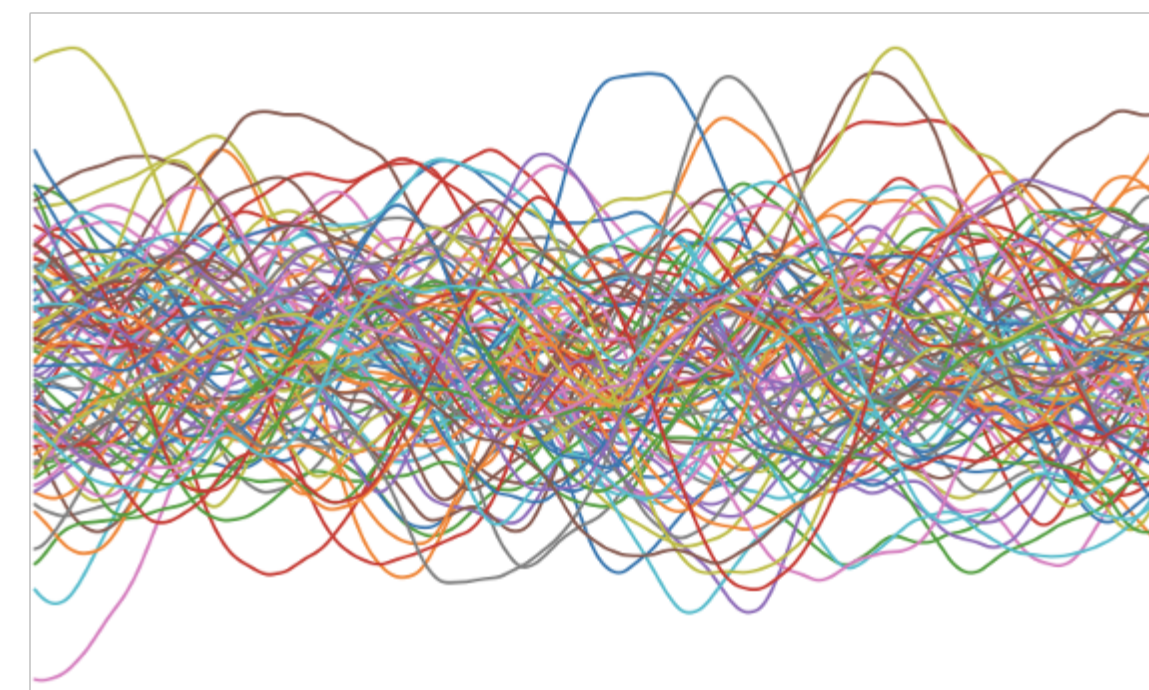
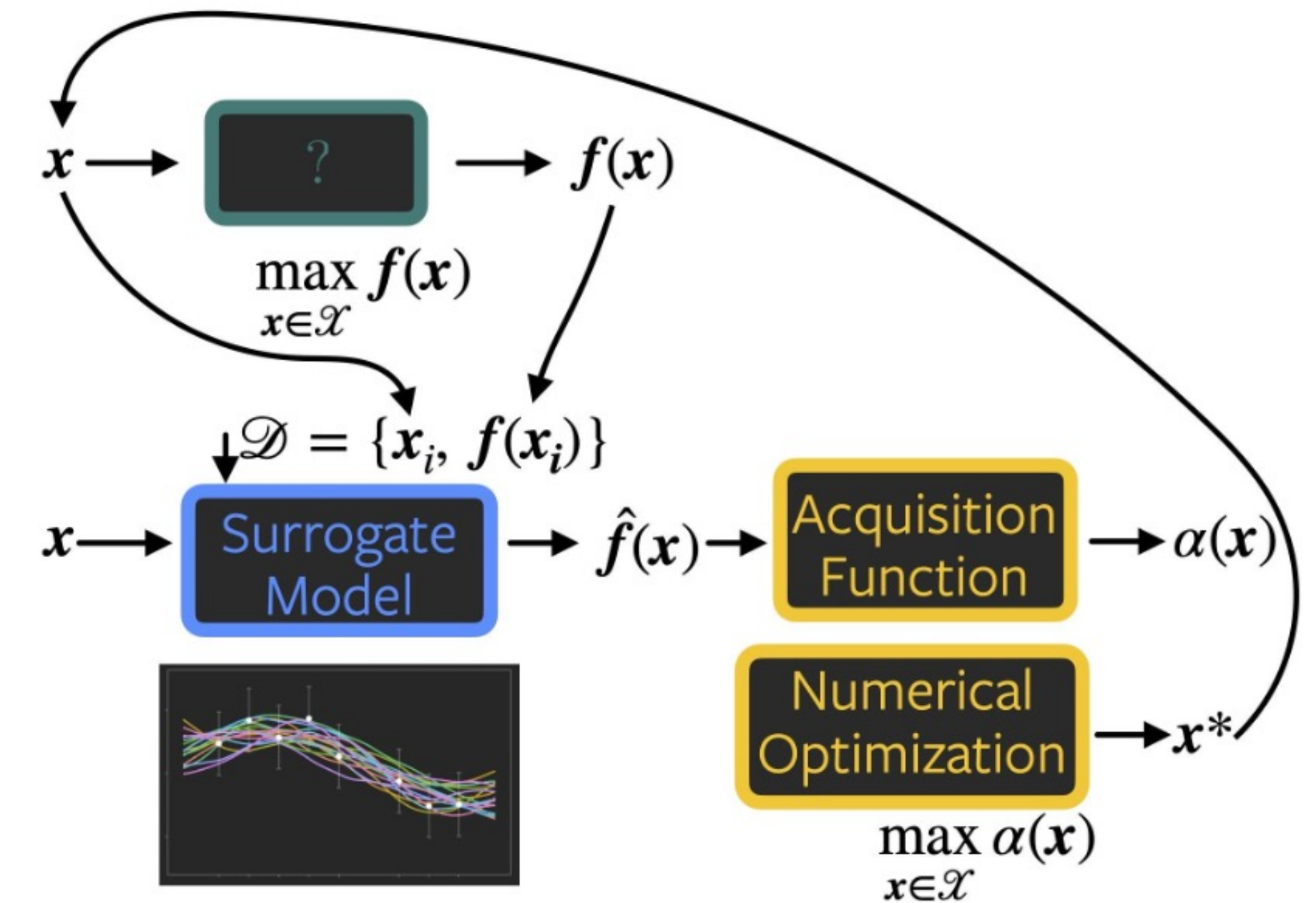
- Defines a distribution over functions:

$$f(x) \sim \mathcal{GP}(\mu(x), k(x, x'))$$

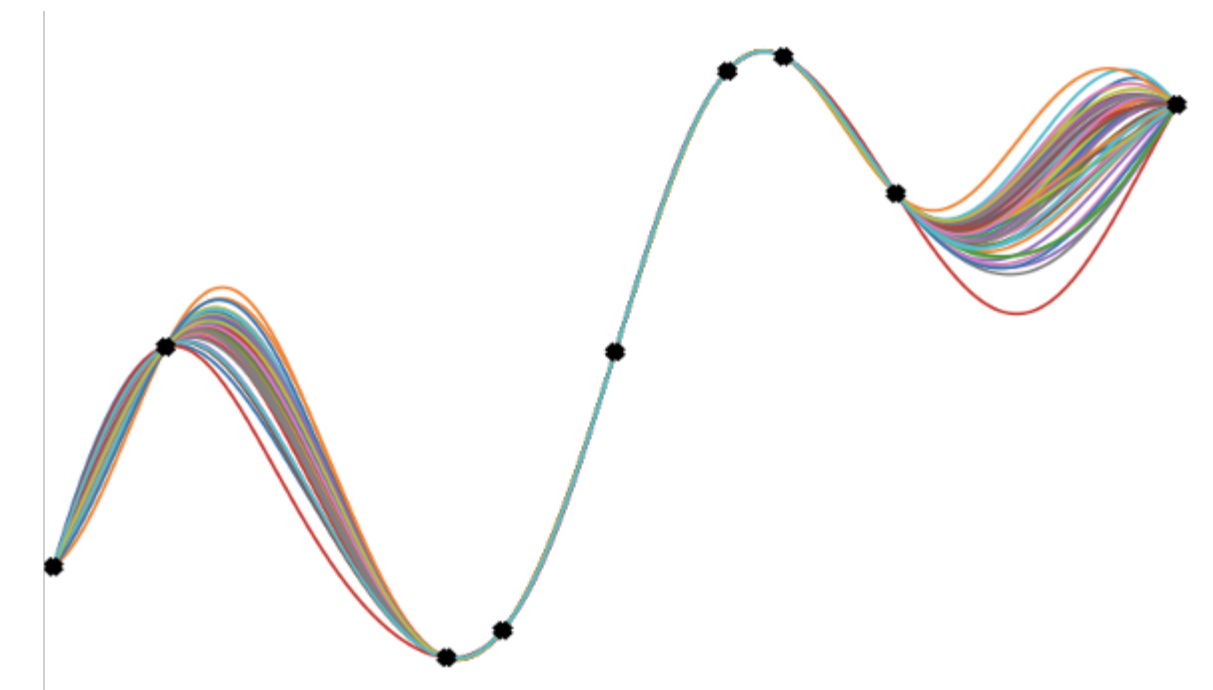
- Covariance function, e.g.

$$k_{SE}(x, x') = s^2 \exp \left(-\frac{1}{2} \sum_{i=1}^d \left(\frac{x_i - x'_i}{\ell_i} \right)^2 \right)$$

lengthscale ℓ_i measures the distance for being uncorrelated along x_i



draws from the prior

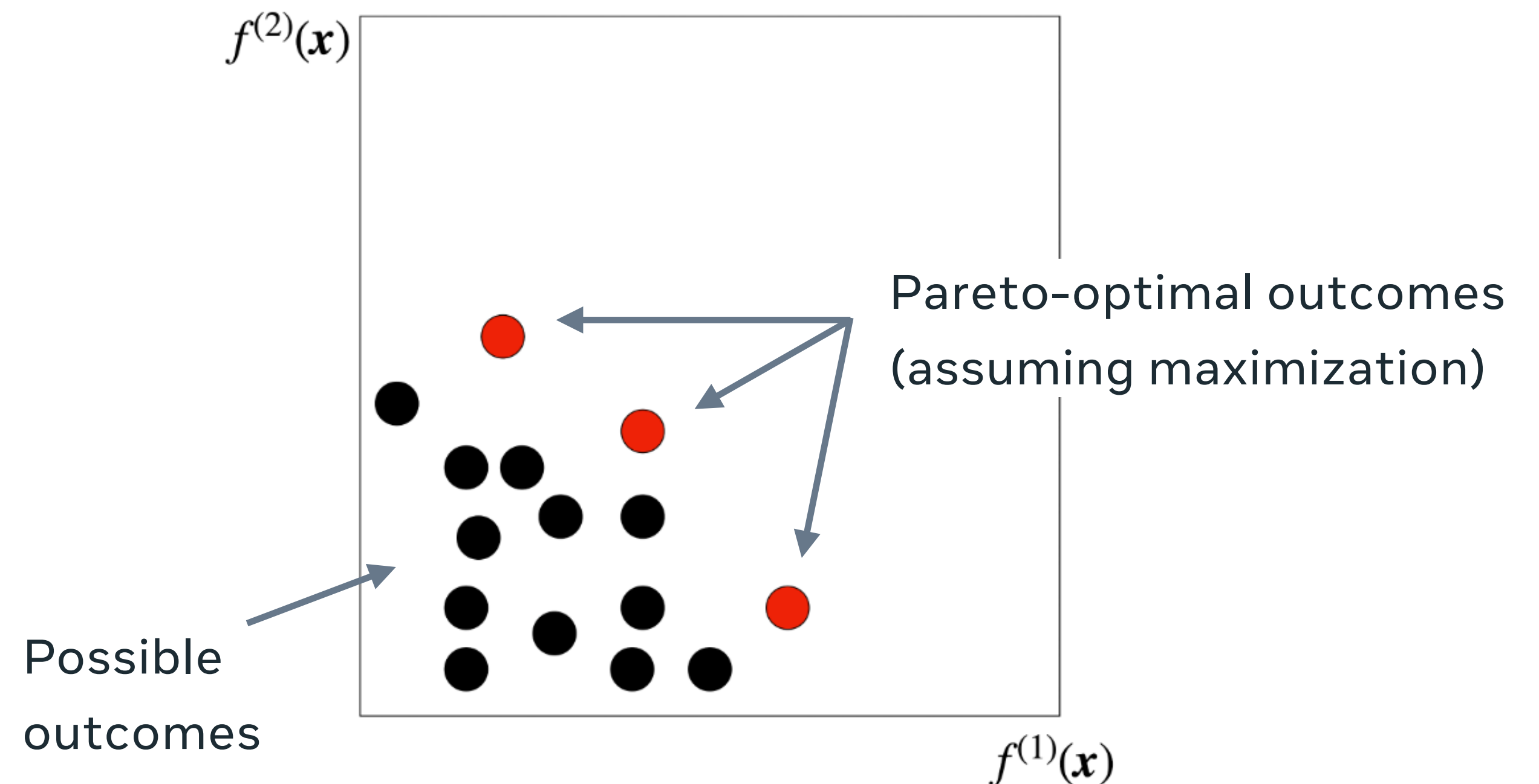


draws from the posterior

Multi-Objective Optimization (MOO)

- Goal: Optimize a **vector-valued black-box function** $f(x) = (f^{(1)}(x), \dots, f^{(m)}(x))$.
 - Typically, no single best solution that simultaneously maximizes all objectives exists.
 - Instead, we identify the **Pareto frontier** of optimal trade-offs
- Various approaches:
 - Domain-specific heuristics
 - Evolutionary algorithms (e.g. NSGA-II)
 - Multi-Objective BO (MOBO)
 - via random scalarizations (e.g. ParEGO)
 - Information-theoretic acquisition functions
 - based on hypervolume improvement

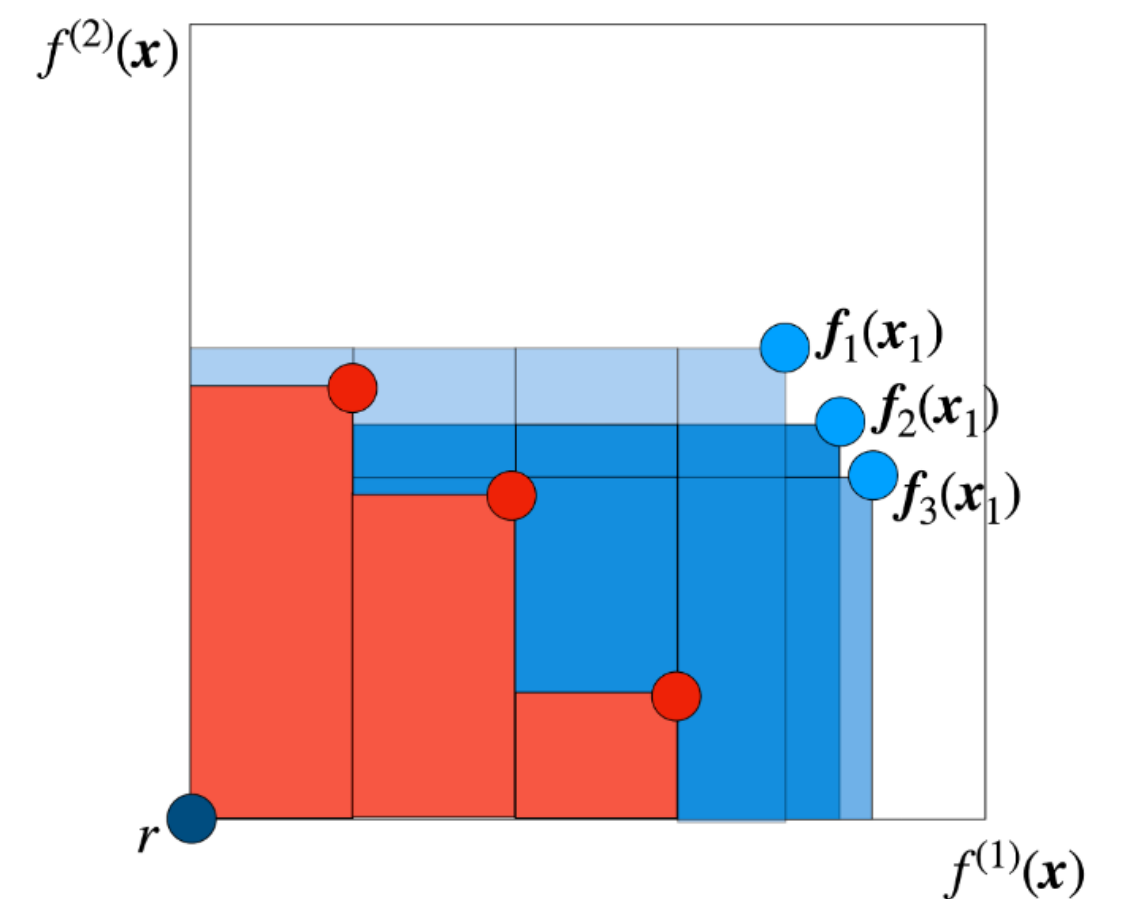
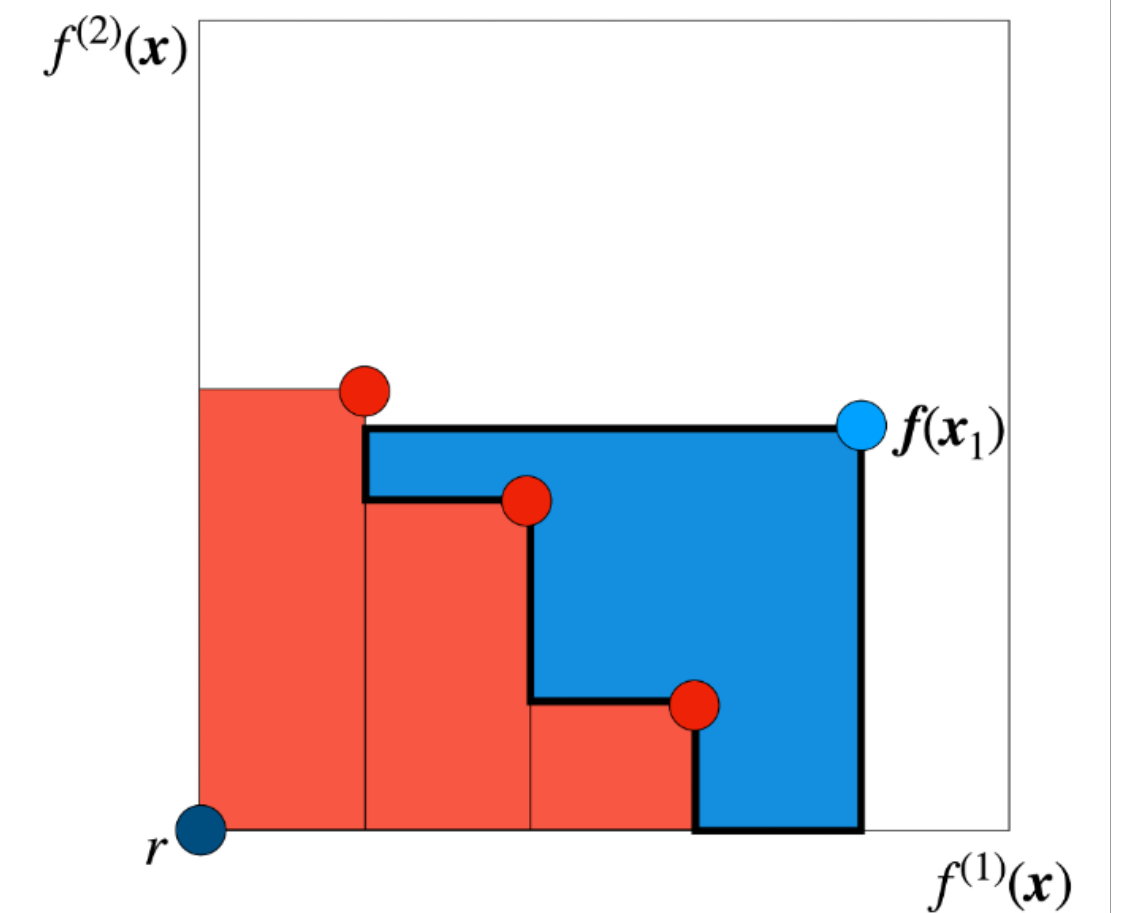
offers superior sample-efficiency



Expected Hypervolume Improvement

- **Hypervolume (HV):** Measure of the quality of a Pareto frontier
 - Volume dominated by the Pareto frontier and bounded by a reference point r
 - Larger HV \rightarrow better Pareto frontier
- **Hypervolume improvement:** increase in HV from a new point or set of points
- **q-Expected Hypervolume Improvement (qEHVI)**
 - One-step Bayes-optimal acquisition function for HV criterion
 - Computes the expected increase in HV under the posterior distribution of the surrogate model
- **q-Noisy Expected Hypervolume Improvement (qNEHVI)**
 - Like qEHVI, but integrates over uncertainty about the value of noisy observations

Default in Ax & BoTorch



3 Applications

3 APPLICATIONS

Application Areas (non-exhaustive)

- **AutoML**

- e.g. Hyperparameter Optimization, Neural Architecture Search, Inference optimization

- **Infrastructure Optimization**

- e.g. tuning parameters of large distributed systems, optimizing fiber backend networks

- Sequential Experimentation in large-scale **online A/B tests**

- **Chemistry & Material Science**

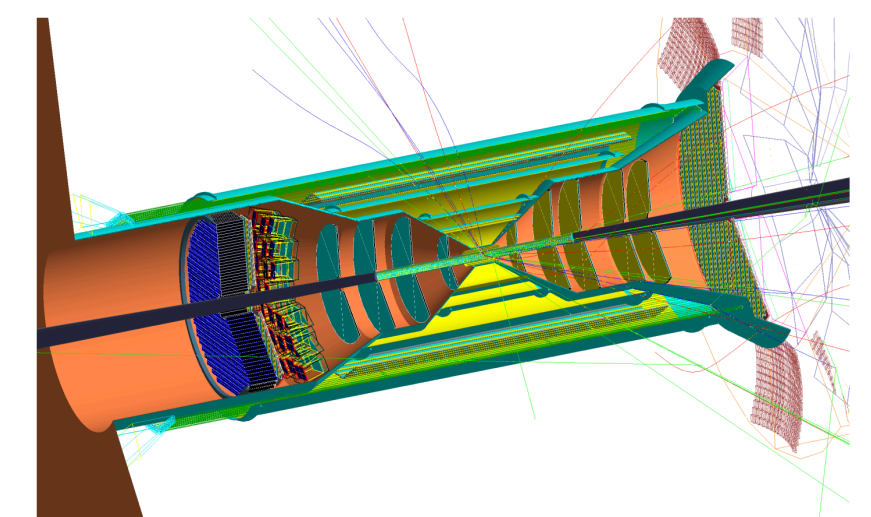
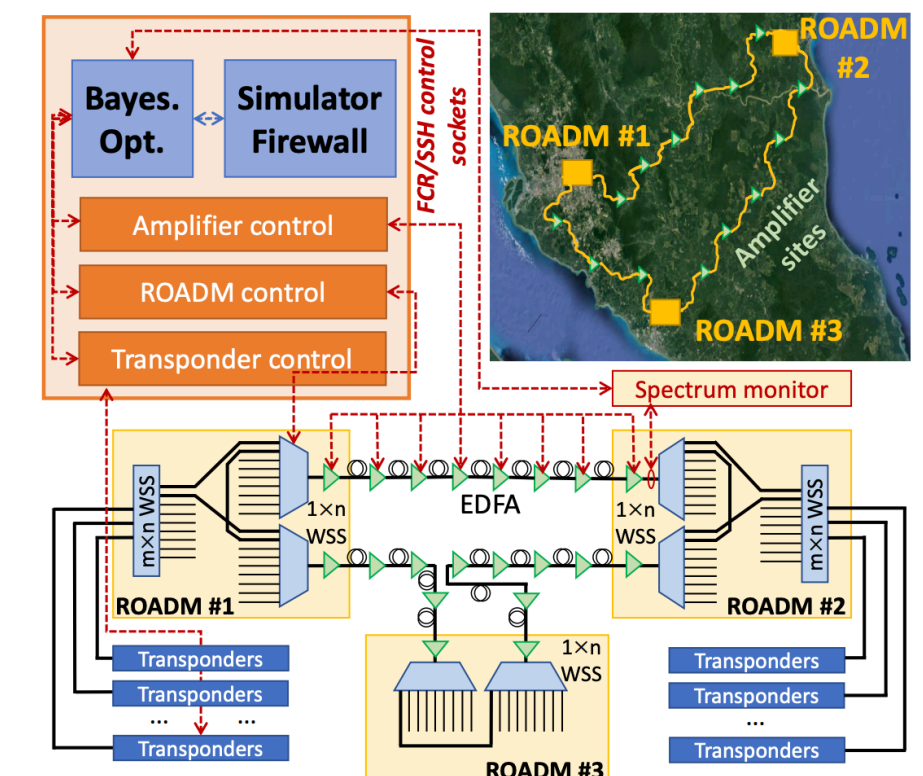
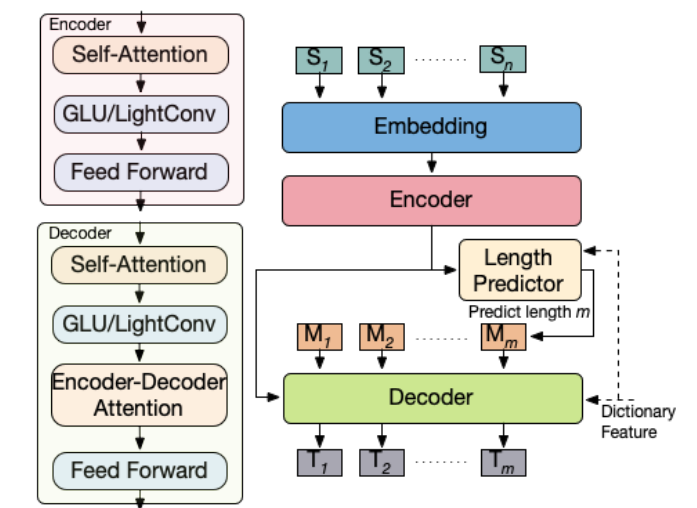
- e.g. process optimization, Automated Chemical Design, material discovery

- **Engineering / Simulation Optimization**

- e.g. design of optical systems for augmented reality (AR) see-through displays

- Health Sciences (Medicine, Neuroscience)

- ...



Research built on Ax & BoTorch

ODBO: BAYESIAN OPTIMIZATION WITH SEARCH SPACE
PRESCREENING FOR DIRECTED PROTEIN EVOLUTION

Bayesian optimization of comprehensive two-dimensional liquid chromatography separations

Jim Boelrijk^{a,d,*}, Bob Pirok^{a,b}, Bernd Ensing^{a,c}, Patrick Forré^{a,d}

^aAI4Science Lab, University of Amsterdam, The Netherlands

^bAnalytical Chemistry Group, Van 't Hoff Institute for Molecular Sciences, University of Amsterdam, The Netherlands

^cComputational Chemistry Group, Van 't Hoff Institute for Molecular Sciences, University of Amsterdam, The Netherlands

^dAMLab Informatics Institute, Universitv of Amsterdam, The Netherlands

GAUCHE: A Library for Gaussian Processes in Chemistry

Ryan-Rhys Griffiths^{*1} Leo Klarner^{*2} Henry B. Moss^{*3} Aditya Ravuri^{*1} Sang Truong^{*4} Bojana Rankovic^{*5}
Yuanqi Du^{*6} Arian Jamash¹ Julius Schwartz¹ Austin Tripp¹ Gregory Kell⁷ Anthony Bourached⁸
Alex J. Chan¹ Jacob Moss¹ Chengzhi Guo¹ Alpha A. Lee¹ Philippe Schwaller⁵ Jian Tang^{9 10 11}

ARTICLE

<https://doi.org/10.1038/s41467-022-28580-6>

OPEN

A self-driving laboratory advances the Pareto front for material properties

Benjamin P. MacLeod^{1,2,5}, Fraser G. L. Parlane^{1,2,5}, Connor C. Rupnow^{1,2,3}, Kevan E. Dettelbach¹, Michael S. Elliott¹, Thomas D. Morrissey^{1,2}, Ted H. Haley¹, Oleksii Proskurin¹, Michael B. Rooney¹, Nina Taherimakhosou¹, David J. Dvorak², Hsi N. Chiu¹, Christopher E. B. Waizenegger¹, Karry Ocean¹, Mehrdad Mokhtari¹ & Curtis P. Berlinguette^{1,2,3,4&}

Tracking Blobs in the Turbulent Edge Plasma of a Tokamak Fusion Device

Woonghee Han^{1,*}, Randall A. Pietersen², Rafael Villamor-Lora², Matthew Beveridge³, Nicola Offeddu⁴, Theodore Golfopoulos¹, Christian Theiler⁴, James L. Terry¹, Earl S. Marmar¹, and Iddo Drori³

¹MIT Plasma Science and Fusion Center, Cambridge, Massachusetts 02139, USA

²MIT, Civil and Environmental Engineering, Cambridge, Massachusetts 02139, USA

³MIT, Electrical Engineering and Computer Science, Cambridge, Massachusetts 02142, USA

⁴École Polytechnique Fédérale de Lausanne (EPFL), Swiss Plasma Center (SPC), CH-1015 Lausanne, Switzerland

*harryhan@mit.edu

POLYHYMNIA Mood – Empowering people to cope with depression through music listening

Eduardo Coutinho
Applied Music Research Lab
University of Liverpool
United Kingdom
e.coutinho@liverpool.ac.uk

Jacopo de Berardinis
School of Computer Science
University of Manchester
United Kingdom
jacopo.deberardinis@manchester.ac.uk

Ayesh Alshukri
Oxford Clinical Trials Research Unit
University of Oxford
United Kingdom
ayesh.alshukri@ndorms.ox.ac.uk

Chris Dowrick
Department of Primary Care and Mental Health
University of Liverpool
United Kingdom
cfd@liverpool.ac.uk

Adaptive nonparametric psychophysics

Lucy Owen
Dartmouth College²

Benjamin Letham
Facebook

Chase Tymms
Facebook Reality Labs

Jonathan Browder
Facebook Reality Labs

Gideon Stoeck
Facebook Reality Labs

Michael Shvartsman
Facebook Reality Labs

High-dimensional Automated Radiation Therapy Treatment Planning via Bayesian Optimization

Qingying Wang^{1,2, a}, Ruoxi Wang^{1, a}, Jiacheng Liu^{1,2}, Fan Jiang¹, Haizhen Yue¹, Yi Du¹, Hao Wu^{1,2,b}

¹ Key Laboratory of Carcinogenesis and Translational Research (Ministry of Education/Beijing), Department of Radiation Oncology, Beijing Cancer Hospital & Institute, Beijing, China

² Institute of Medical Technology, Peking University Health Science Center

Advanced Operating Conditions Search applied in Analog Circuit Verification

Cristian Manolache¹, Alexandru Caranica¹, Marius Stănescu¹, Horia Cucu¹, Andi Buzo², Cristian Diaconu², Georg Pelz²

¹University “Politehnica” of Bucharest, Romania

²Infineon Technologies, Munich, Germany

{cristian.manolache, alexandru.caranica, adrian.stanescu, horia.cucu}@upb.ro
{andi.buzo, cristian-vasile.diaconu, georg.pelz}@infineon.com

Noisy Bayesian optimization for variational quantum eigensolvers

Giovanni Iannelli^{a,b,c,d,*} and Karl Jansen^a

ARTICLE

<https://doi.org/10.1038/s41467-021-25757-3>

OPEN

Turn-key constrained parameter space exploration for particle accelerators using Bayesian active learning

Ryan Roussel^{1&}, Juan Pablo Gonzalez-Aguilera¹, Young-Kee Kim¹, Eric Wisniewski², Wanming Liu², Philippe Piot^{2,3}, John Power², Adi Hanuka⁴ & Auralee Edelen⁴

Characterization of Exercise-Induced Myocardium Growth Using Finite Element Modeling and Bayesian Optimization

Yiling Fan^{1,2,3}, Jaume Coll-Font^{1,4,5}, Maaïke van den Boomen^{1,4,5}, Joan H. Kim¹, Shi Chen¹, Robert Alan Eder¹, Ellen T. Roche^{2,3,5*} and Christopher T. Nguyen^{1,4,5*}

¹ Cardiovascular Bioengineering and Imaging Laboratory, Cardiology Division, Massachusetts General Hospital, Charlestown, MA, United States; ² Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, United States; ³ Institute for Medical Engineering and Science, Massachusetts Institute of Technology, Cambridge, MA, United States; ⁴ Athinoula A. Martinos Center for Biomedical Imaging, Charlestown, MA, United States; ⁵ Harvard Medical School, Boston, MA, United States

Rapidly Inferring Personalized Neurostimulation Parameters with Meta-Learning: A Case Study of Individualized Fiber Recruitment in Vagus Nerve Stimulation

Ximeng Mao^{1,2}, Yao-Chuan Chang³, Stavros Zanos³, Guillaume Lajoie^{1,4,5,*}

Active Bayesian Causal Inference

Christian Toth
TU Graz

Lars Lorch
ETH Zürich

Christian Knoll
TU Graz

Andreas Krause
ETH Zürich

Franz Pernkopf
TU Graz

Robert Peharz^{*}
TU Graz

Julius von Kügelgen^{*}
MPI for Intelligent Systems, Tübingen
University of Cambridge

Bayesian optimization of distributed neurodynamical controller models for spatial navigation

Armin Hadzic^{a,*}, Grace M. Hwang^{a,b,1}, Kechen Zhang^c, Kevin M. Schultz^a, Joseph D. Monaco^c

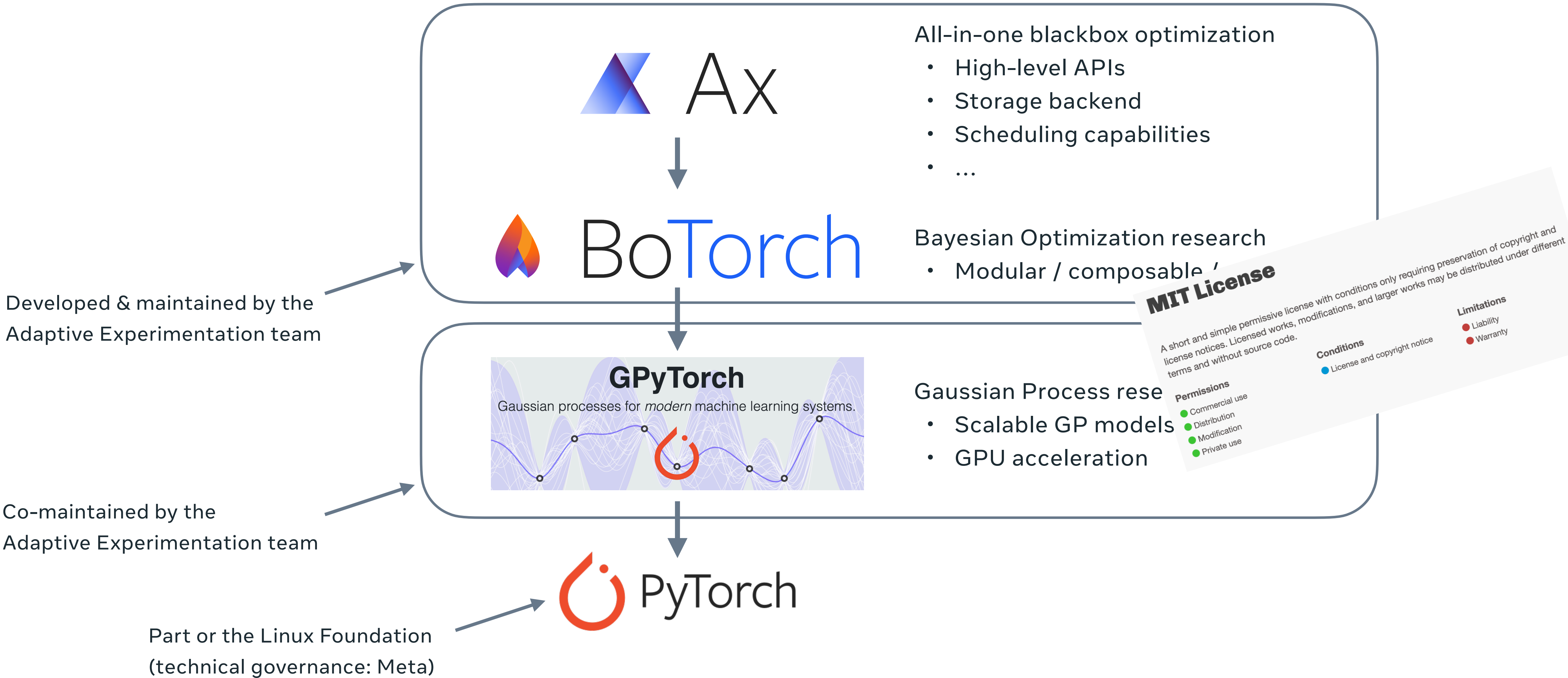
Article

Lost in Optimization of Water Distribution Systems: Better Call Bayes

Antonio Candelieri^{1,*}, Andrea Ponti^{2,3}, Ilaria Giordani^{2,4} and Francesco Archetti^{3,4}

4 Tutorial: Hands-on Examples

The Ax & BoTorch Tech Stack



Ax APIs: Service

- Ask-tell interface
- **Simple yet powerful**
- Scheduling & Evaluation of trials is responsibility of the user
- **Should be the go-to** if additional advanced scheduling / deployment functionalities are not required

```
from ax.service.ax_client import AxClient
from ax.utils.measurement.synthetic_functions import branin
```

```
ax_client = AxClient()
ax_client.create_experiment(
    name="branin_test_experiment",
    parameters=[
        {
            "name": "x1",
            "type": "range",
            "bounds": [-5.0, 10.0],
        },
        {
            "name": "x2",
            "type": "range",
            "bounds": [0.0, 10.0],
        },
    ],
    objective_name="branin",
    minimize=True,
)
```

(synthetic) black-box function f

Inferred to be “float”.
Can also define integer or categorical parameters.

```
for _ in range(15):
    parameters, trial_index = ax_client.get_next_trial()
    result = branin(parameters["x1"], parameters["x2"])
    ax_client.complete_trial(
        trial_index=trial_index,
        raw_data=result,
    )

best_parameters, metrics = ax_client.get_best_parameters()
```

Here is where we
evaluate f

4 HANDS-ON EXAMPLES

Ax APIs: Developer

- Highly flexibly and customizable (thus more complex)
- Runner and Metric abstractions allow implementing **trial deployment / management** and **data fetching** capabilities for **different backends**

```
from ax import *  
  
class MockRunner(Runner):  
    def run(self, trial):  
        return {"name": str(trial.index)}
```

Usually used to deploy to backend (e.g. simulation cluster)

```
search_space = SearchSpace(  
    parameters=[  
        RangeParameter(  
            name="x1", parameter_type=ParameterType.FLOAT, lower=-5, upper=10  
        ),  
        RangeParameter(  
            name="x2", parameter_type=ParameterType.FLOAT, lower=0, upper=15  
        ),  
    ]  
)  
exp = Experiment(  
    name="test_branin",  
    search_space=search_space,  
    optimization_config=OptimizationConfig(  
        objective=Objective(  
            metric=BraninMetric(name="branin", param_names=["x1", "x2"]),  
            minimize=True,  
        ),  
    ),  
    runner=MockRunner(),  
)
```

Example synthetic metric.

Quasi-random
initial exploration

```
sobol = Models.SOBOL(exp.search_space)  
for _ in range(5):  
    trial = exp.new_trial(generator_run=sobol.gen(n=1))  
    trial.run()  
    trial.mark_completed()
```

GP model + Expected
Improvement

```
for _ in range(15):  
    gp_ei = Models.GPEI(experiment=exp, data=exp.fetch_data())  
    generator_run = gp_ei.gen(n=1)  
    best_arm, _ = generator_run.best_arm_predictions  
    trial = exp.new_trial(generator_run=generator_run)  
    trial.run()  
    trial.mark_completed()
```

```
exp.fetch_data()  
best_parameters = best_arm.parameters
```


Ax APIs: Scheduler

- Orchestrates **fully automated closed-loop optimization** in a **backend-agnostic fashion** (leveraging `Metric` and `Runner` implementations)
- Supports advanced scheduling capabilities including
 - Parallel trial evaluations
 - Fully asynchronous trial evaluations
 - Early stopping of trials that provide partial results during the run
 - Early stopping of the overall optimization based on customizable stopping strategies

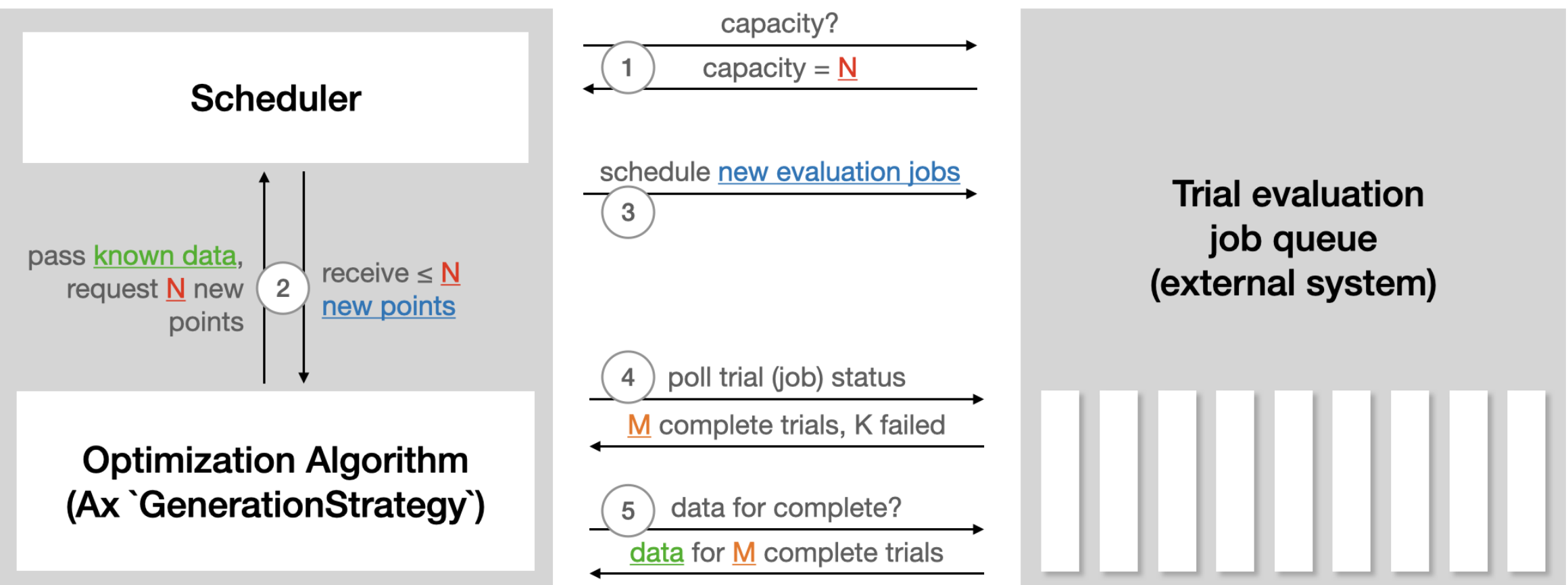
```
from ax import *
from ax.modelbridge.generation_strategy importGenerationStrategy
from ax.service import Scheduler

# Full `Experiment` and `GenerationStrategy` instantiation
# omitted for brevity, refer to tutorials for detail.
experiment = Experiment(...)
generation_strategy = GenerationStrategy(...)

scheduler = Scheduler(
    experiment=experiment,
    generation_strategy=generation_strategy,
    options=SchedulerOptions(), # Configurations for how to run the experiment
)


scheduler.run_n_trials(100) # Automate running 100 trials and reporting results
```

← can be chosen automatically



Getting Started: Multi-Objective BO w/ Ax

https://ax.dev/tutorials/multiobjective_optimization.html

Ax

DocsTutorialsAPIGitHubSearch

v: stable

Tutorials

- Overview

API Comparison

- Loop API
- Service API
- Developer API

Deep Dives

- Visualizations
- Generation Strategy
- Scheduler
- Modular `BoTorchModel`

Bayesian Optimization

- Hyperparameter Optimization for PyTorch
- Hyperparameter Optimization via Ravg

Multi-Objective Optimization Ax API

Using the Service API

For Multi-objective optimization (MOO) in the `AxClient`, objectives are specified through the `ObjectiveProperties` dataclass. An `ObjectiveProperties` requires a boolean `minimize`, and also accepts an optional floating point `threshold`. If a `threshold` is not specified, Ax will infer it through the use of heuristics. If the user knows the region of interest (because they have specs or prior knowledge), then specifying the thresholds is preferable to inferring it. But if the user would need to guess, inferring is preferable.

To learn more about how to choose a threshold, see [Set Objective Thresholds to focus candidate generation in a region of interest](#). See the [Service API Tutorial](#) for more information on running experiments with the Service API.

In `[]`:


```
from ax.service.ax_client import AxClient
from ax.service.utils.instantiation import ObjectiveProperties

import torch
```



Under the Hood: Multi-Objective BO w/ BoTorch

https://botorch.org/tutorials/multi_objective_bo



[Docs](#) [Tutorials](#) [API Reference](#) [Papers](#) [GitHub](#) [Search](#) [v: stable](#)

Tutorials

Overview

Using BoTorch with Ax

Using a custom BoTorch model

Writing a custom acquisition function

Full Optimization Loops

q-Noisy Constrained EI

Bayesian optimization with pairwise comparison data

Bayesian optimization with preference exploration

Trust Region Bayesian Optimization (TuRBO)

Scalable Constrained Bayesian Optimization (SCBO)

Noisy, Parallel, Multi-Objective BO in BoTorch with qEHVI, qNEHVI, and qNParEGO

In this tutorial, we illustrate how to implement a simple multi-objective (MO) Bayesian Optimization (BO) closed loop in BoTorch.

In general, we recommend using [Ax](#) for a simple BO setup like this one, since this will simplify your setup (including the amount of code you need to write) considerably. See [here](#) for an Ax tutorial on MOBO. If desired, you can use a custom BoTorch model in Ax, following the [Using BoTorch with Ax](#) tutorial. Given a `MultiObjective`, Ax will default to the `qNEHVI` acquisition function. If desired, this can also be customized by adding `"botorch_acqf_class": <desired_botorch_acquisition_function_class>`, to the `model_kwargs`.

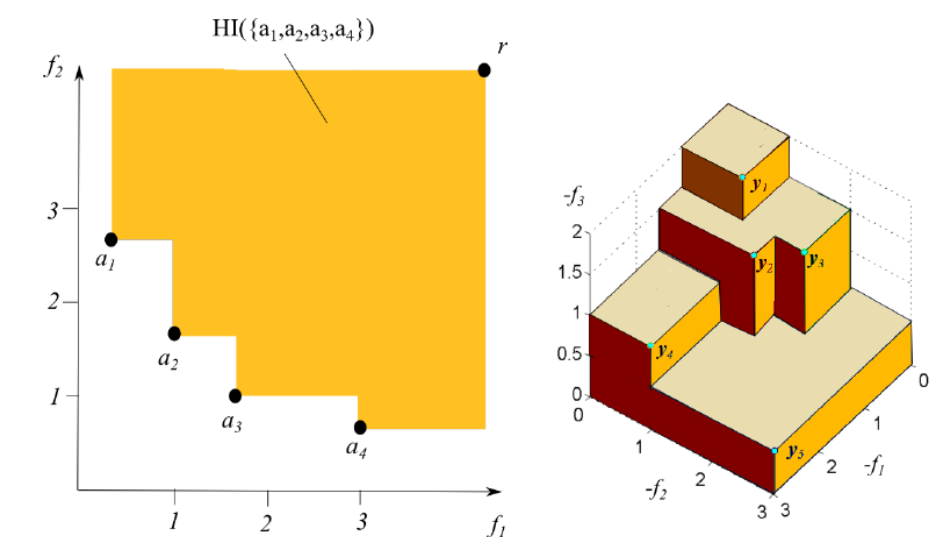
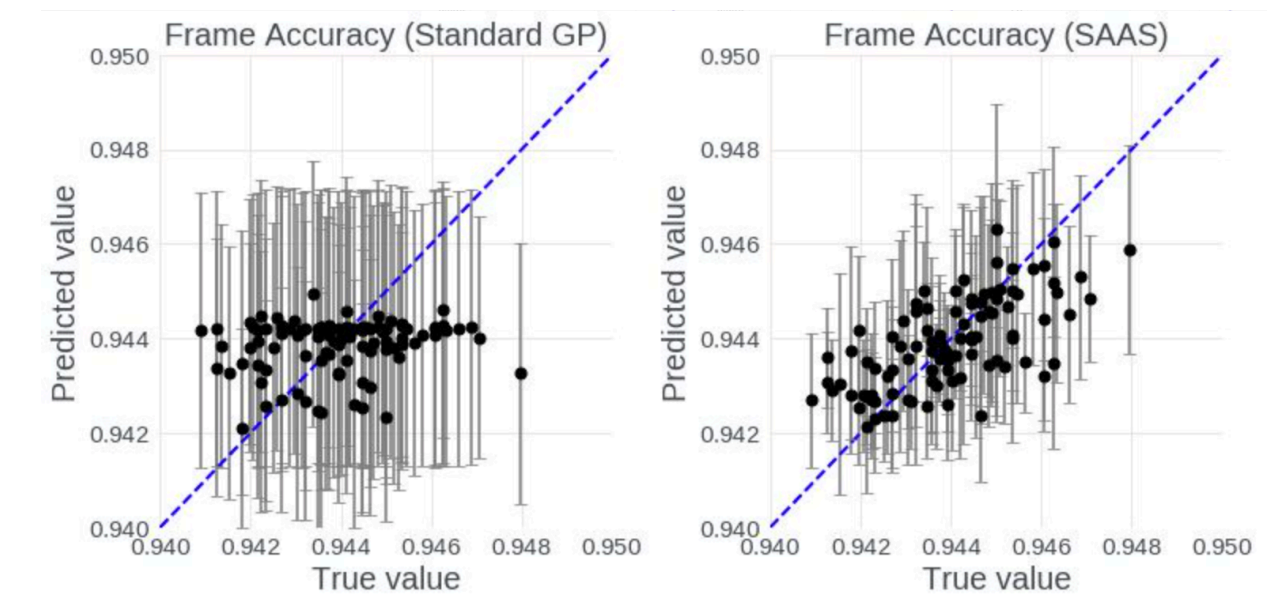
We use the parallel ParEGO (`qParEGO`) [1], parallel Expected Hypervolume Improvement (`qEHVI`) [1], and parallel Noisy Expected Hypervolume Improvement (`qNEHVI`) [2] acquisition functions to optimize a synthetic BraninCurrin problem test function with additive Gaussian observation noise over a 2-parameter search space $[0,1]^2$. See `botorch/test_functions/multi_objective.py` for details on BraninCurrin. The noise standard deviations are 15.19 and 0.63 for each objective, respectively.

Since botorch assumes a maximization of all objectives, we seek to find the Pareto frontier, the set of



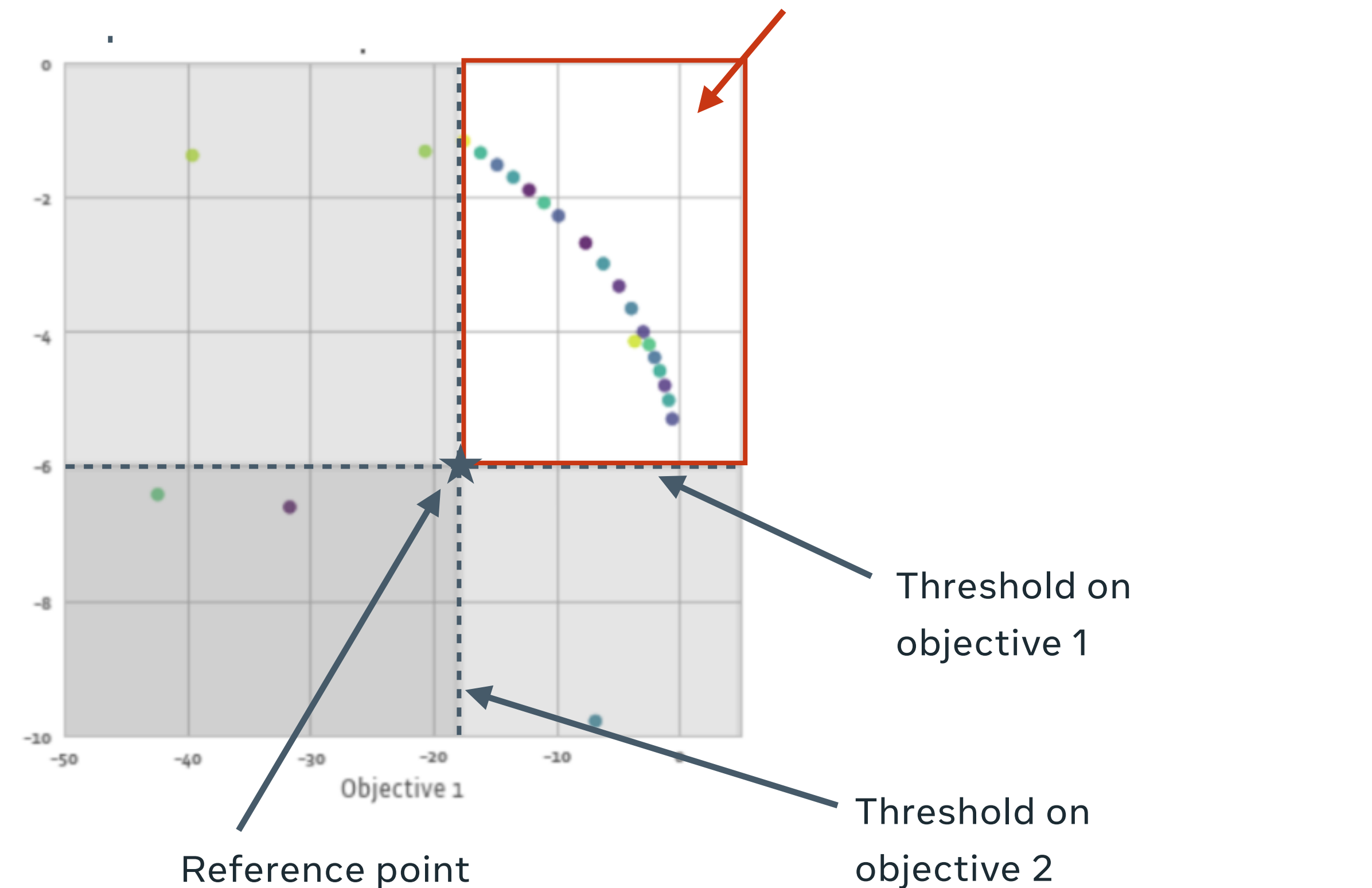
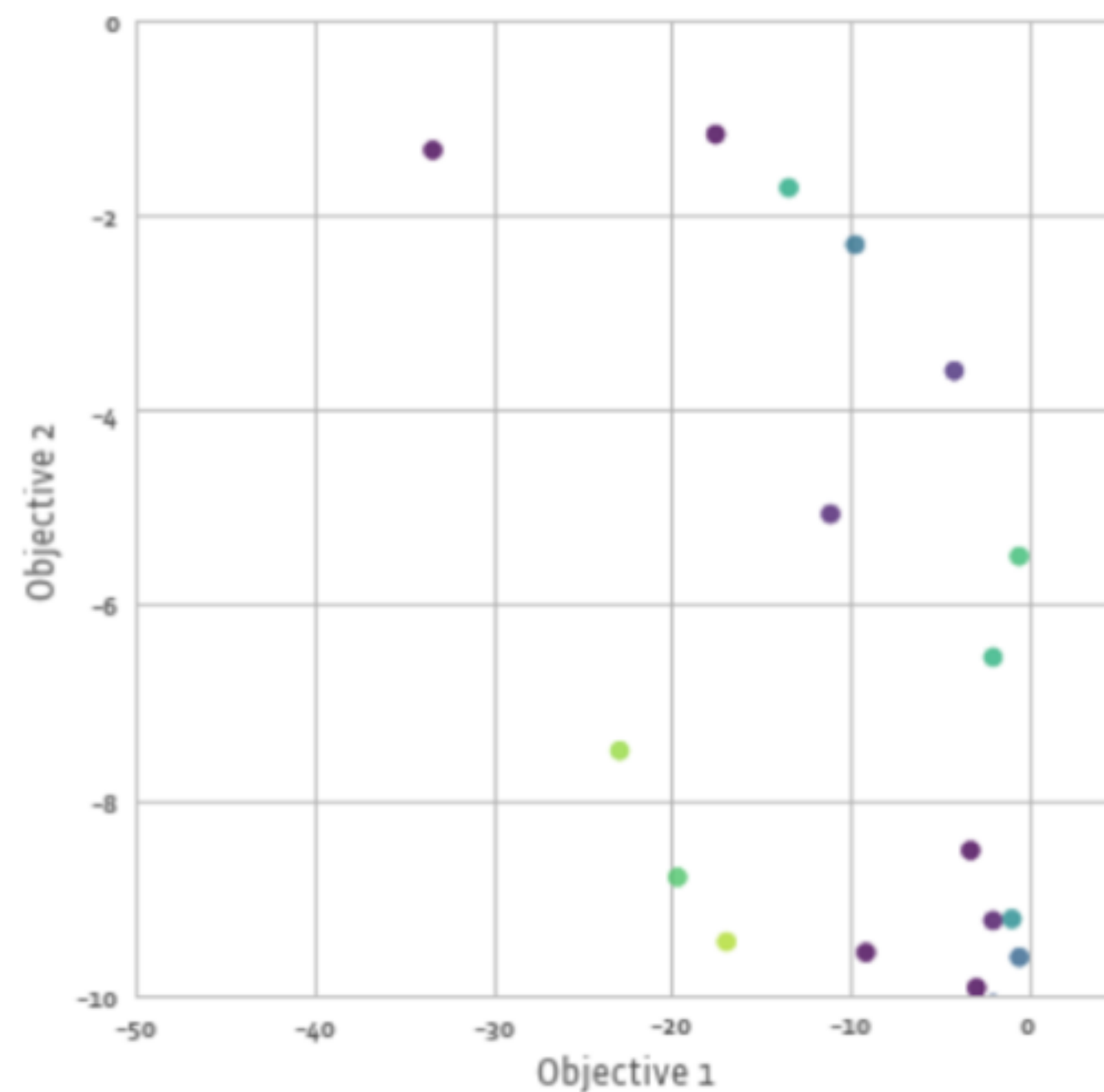
Common Gotchas: Scalability

- In the number of data points / function **evaluations (n)**:
 - Model fitting is $O(n^3)$ with standard GP models / algorithms \Rightarrow typically limited to a few hundred observations (more scalable models / methods exist)
- In the dimensional of the search space / number of **parameters (d)**:
 - Computational scaling is relatively benign (kernel method!)
 - But: Statistical efficiency & model quality becomes an issue!
 - SAASBO (sparsity-inducing prior) and TuRBO (trust-region approach) can help scale to hundreds of parameters
- In the number of **objectives (m)** for MOO:
 - Computing HV is super-polynomial in $m \Rightarrow$ does not scale to more than 3-4 objectives (scaling in outcome constraints on other outcomes is more graceful)
- Random scalarization (e.g. qParEGO) can work with larger m (though less sample-efficient)



Common Gotchas: Setting Objective Thresholds

- Objective thresholds (Ax) = Reference point (BoTorch)
- These **define the “area of interest”** to the MOO algorithm **in the outcome space**
- Setting these properly is important for the efficiency of the optimization



Other Common Gotchas

Noisy observations

- In practice, outcomes are often subject to **observation noise** (either known or unknown)
 - If noise level is unknown, is important for the model to understand if it needs to **infer a noise level** (in Ax that means passing a NaN/None variance observation, in BoTorch it means using a `SingleTaskGP` or similar)
 - If noise level is known, providing it to model can be very helpful

Numerical precision

- Inference with GPs often involves solving ill-conditioned linear systems
 - This is especially problematic if there are many repeated or very similar design points
 - Always **work in double precision if possible!**

5 Advanced Features

High-Dimensional Multi-Objective BO

Limitations of existing Multi-Objective Bayesian Optimization methods (e.g. qEHVI):

- Poor performance of GP models in **high-dimensional domains** ($\gtrsim 15$ parameters)
- May not scale well to the **large-batch / high-throughput setting** with many observations
- Often do not support **black-box constraints**

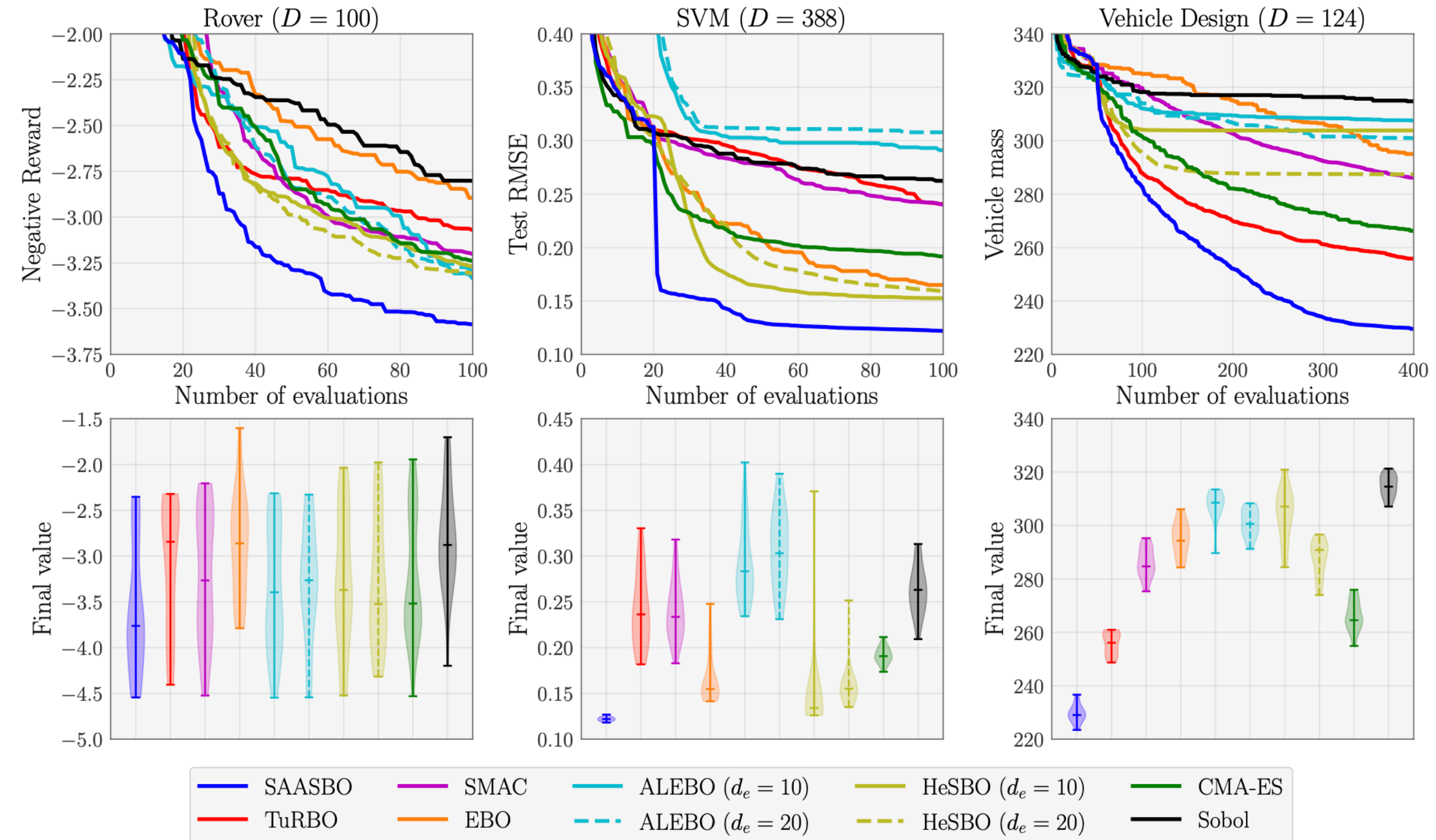
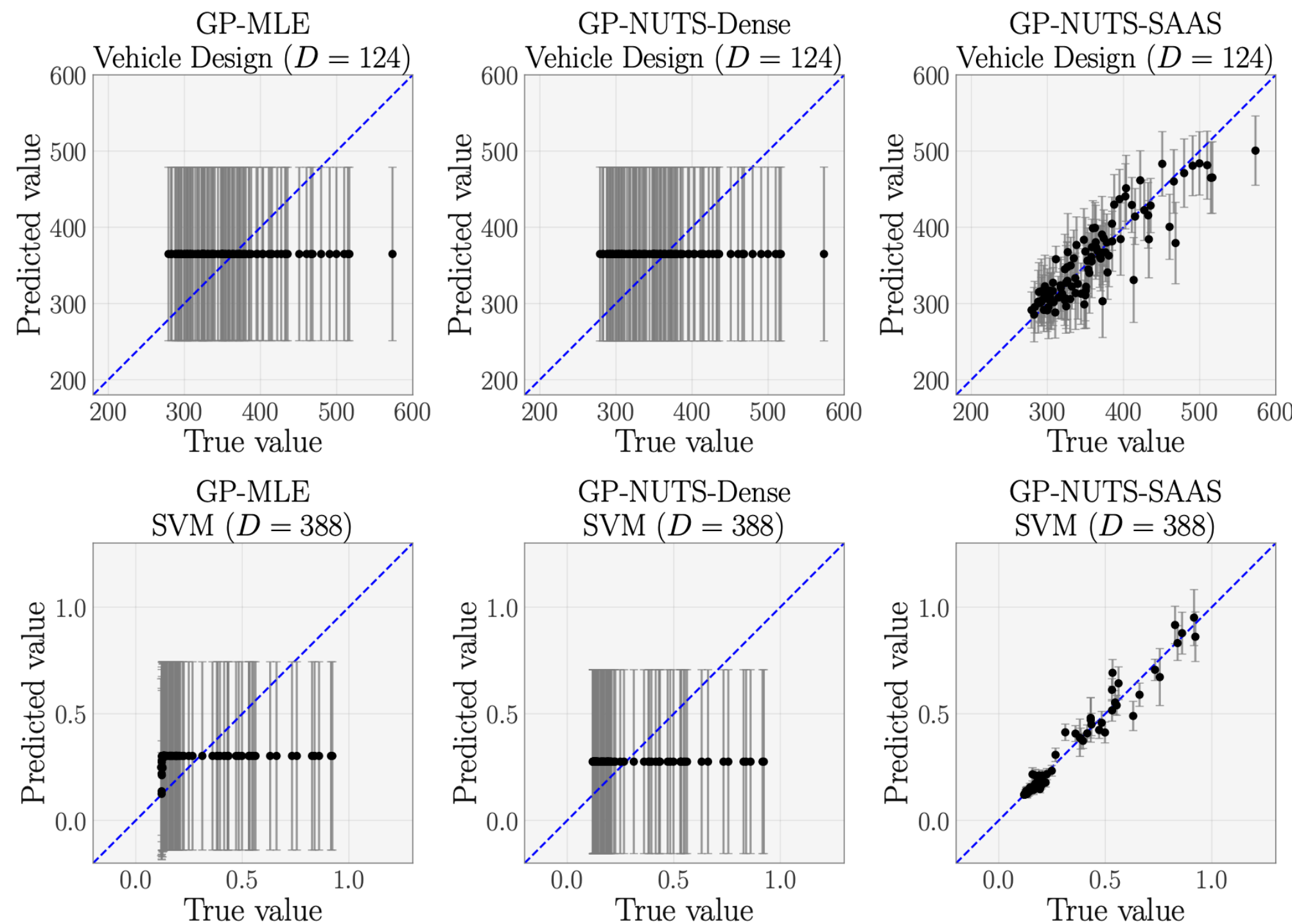
Our contributions:

- **SAASBO**: “Sparse Axis-Aligned Subspace Bayesian Optimization”
 - Sparsity-inducing prior on the kernel lengthscales **identifies the most important parameters**
 - Allows model to scale to **hundreds of dimensions**
- **MORBO**: “Multi-Objective trust Region Bayesian Optimization”
 - The first scalable MO-BO algorithm that is **practical for high-dim problems w/ thousands of evaluations**
 - **Efficient scaling** in both dimension and *#* of evaluations; **well-distributed high-quality Pareto frontiers**

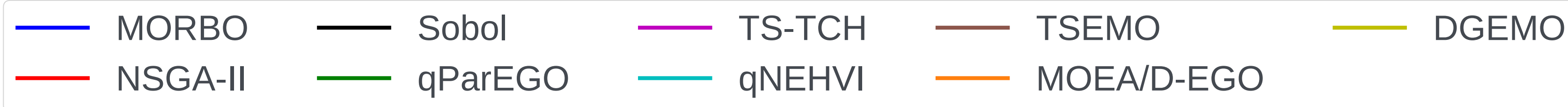
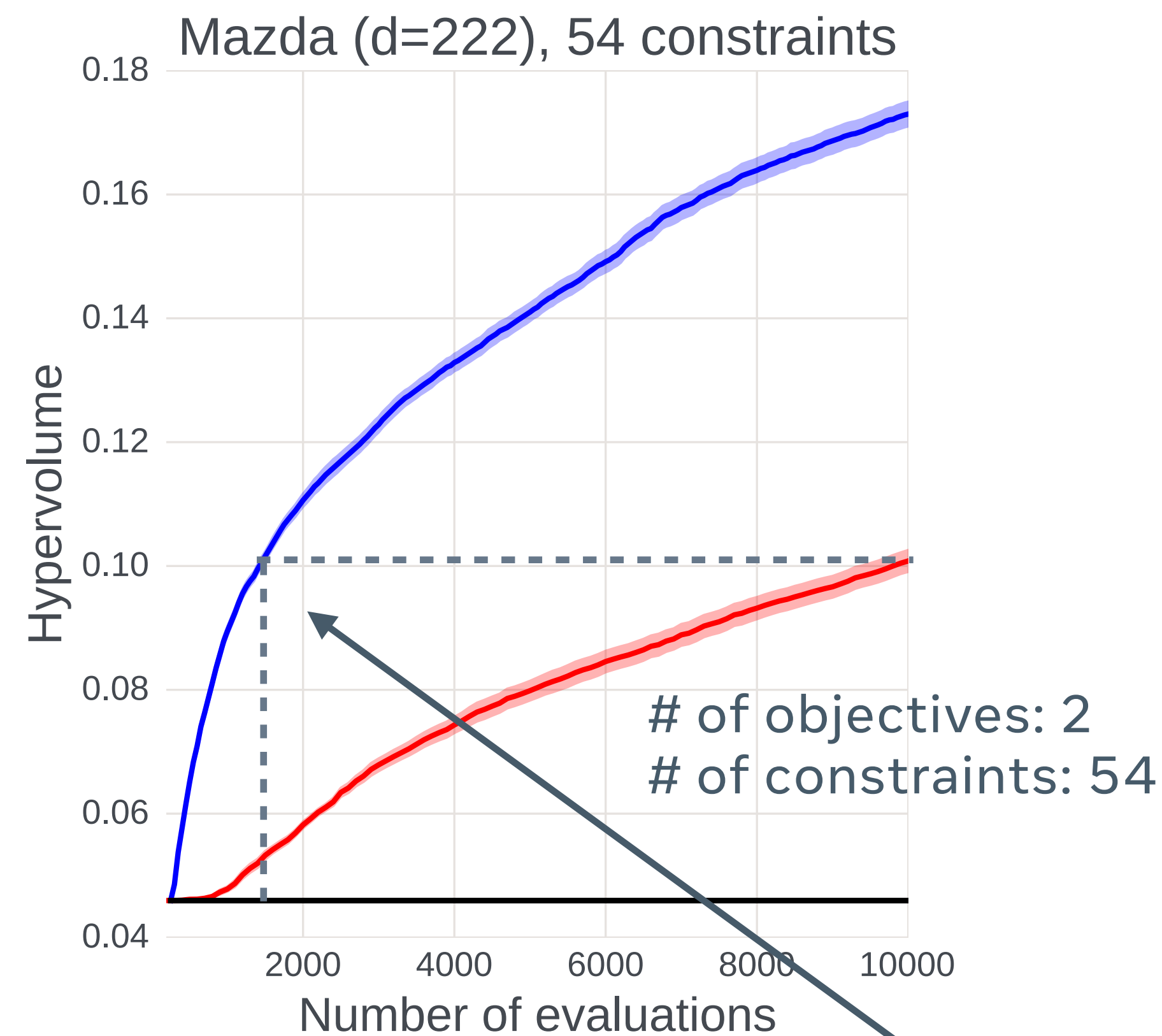
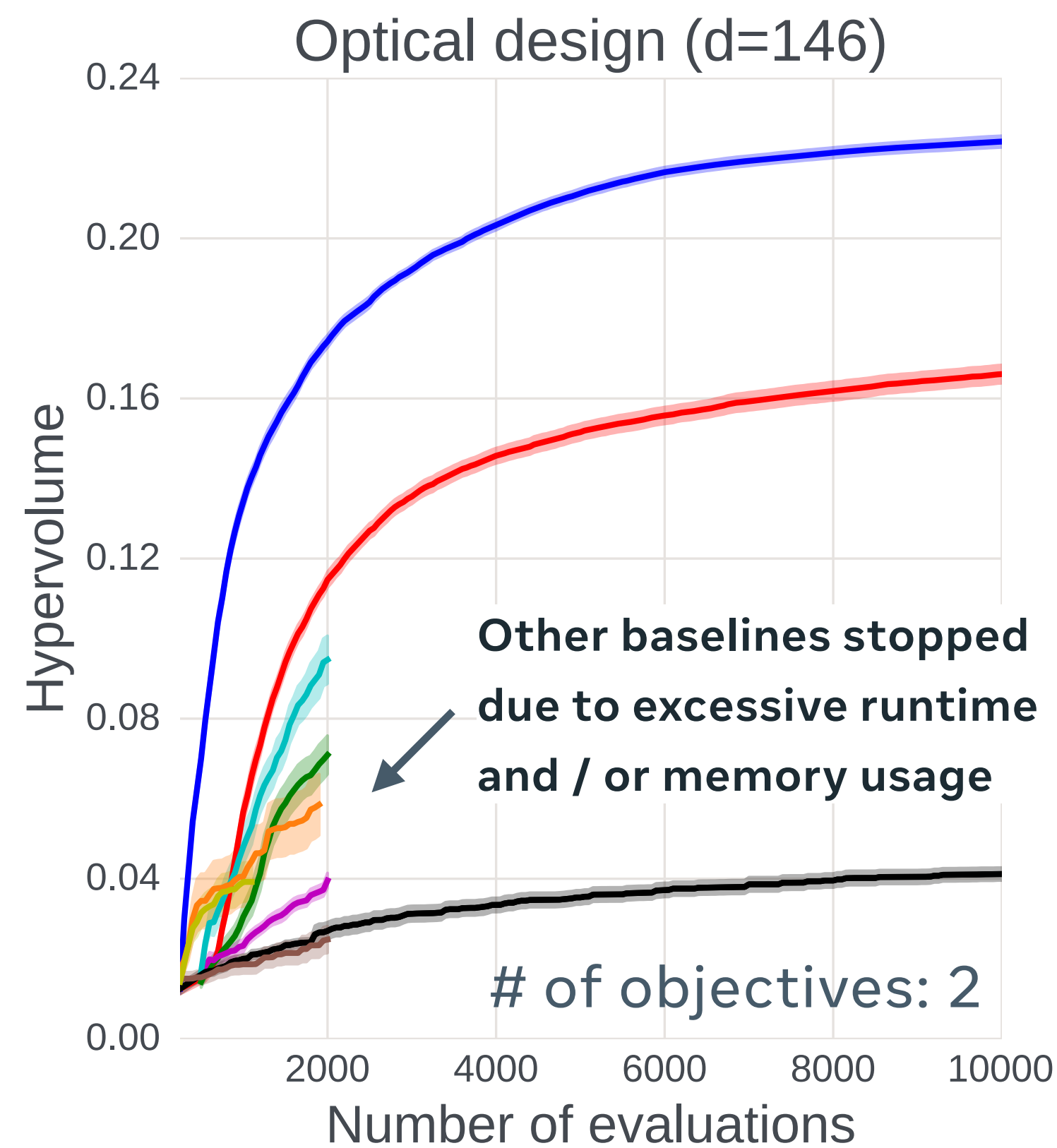
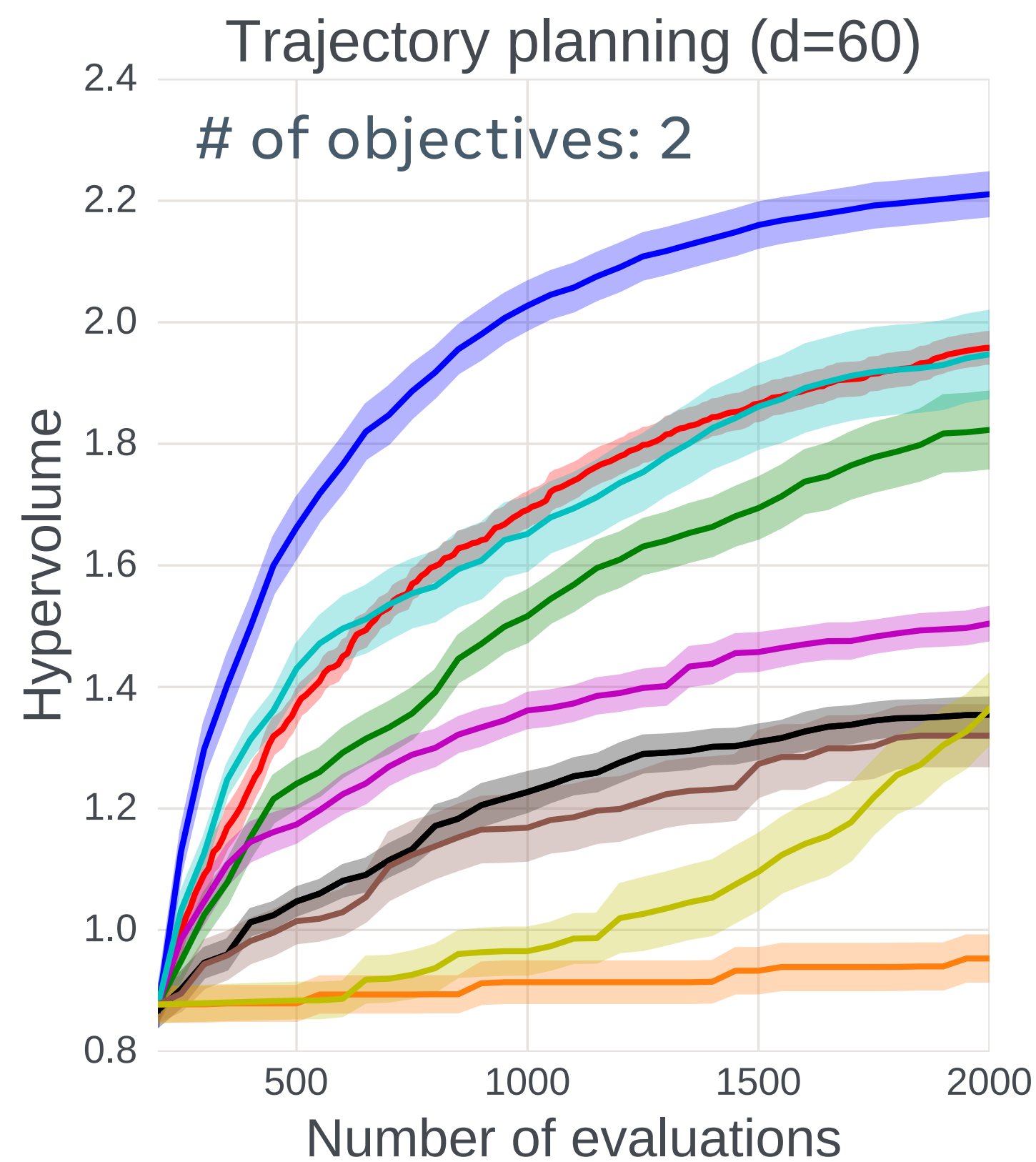
SAASBO: Improved Surrogate Modeling

Available in Ax
& BoTorch!

- In many high-dimensional problems, only a rather small set of the parameters have outsized influence
- **Idea:** Place a sparsity inducing prior over the length scales of the GP kernel
 - Inference performed via MCMC (NUTS)
- Can be combined with MOO



MORBO: High-Throughput MOBO



Robust Multi-Objective Optimization

Available in Ax
& BoTorch!

- In practice, we may not be able to realize a design $x \in \mathbb{X}$ exactly
 - e.g. we cannot build 100% to spec due to manufacturing tolerances, environmental conditions, etc.
- How can we be robust to this input noise? How to even formulate this in a multi-objective setting?

Definition 4.1. Given input noise $\xi \sim P(\xi)$ where $\xi \in \mathbb{R}^d$ and a confidence level $\alpha \in [0, 1]$, the value-at-risk for a given point x is:

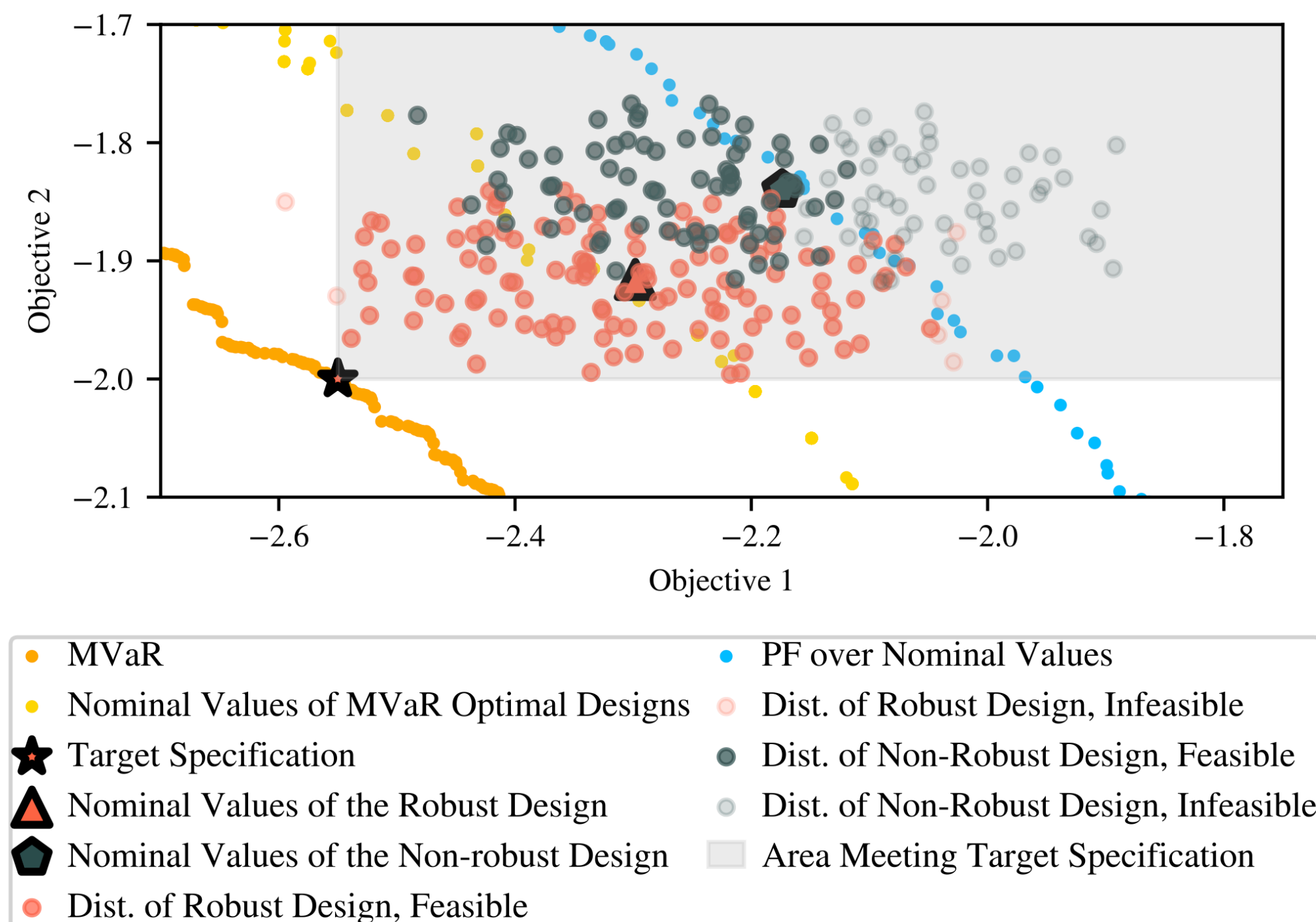
$$\text{VAR}_\alpha[f(x \diamond \xi)] = \sup\{z \in \mathbb{R} : P[f(x \diamond \xi) \geq z] \geq \alpha\}.$$

Definition 4.2. The MVAR of f for a given point x and confidence level $\alpha \in [0, 1]$ is:

$$\text{MVAR}_\alpha[f(x \diamond \xi)] = \text{PARETO}(\{z \in \mathbb{R}^M : P[f(x \diamond \xi) \geq z] \geq \alpha\}).$$

The MVAR set over X specifies objective vectors z such that there exists a known design $x \in X$ with corresponding random objectives $f(x \diamond \xi)$ under $P(\xi)$ that dominate z with probability $\geq \alpha$.

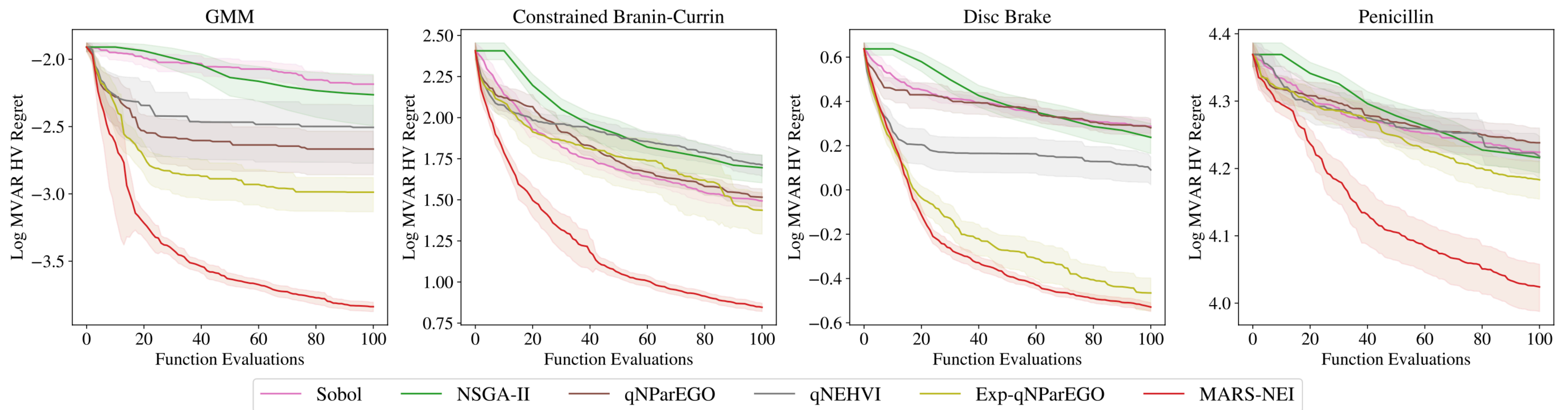
- Contribution: MVAR Approximation based on Random Scalarizations (MARS)



Robust Multi-Objective Optimization

Available in Ax
& BoTorch!

- MARS uses a **novel theoretical connection** between the MVAR and the VAR of a particular scalarization of the objectives
 - This motivates a family of **computationally efficient BO methods for identifying MVAR**
- MARS allows to **identify optimal tradeoffs subject to a constraint on the yield** (% of feasible designs)



Mixed Discrete & Continuous Spaces

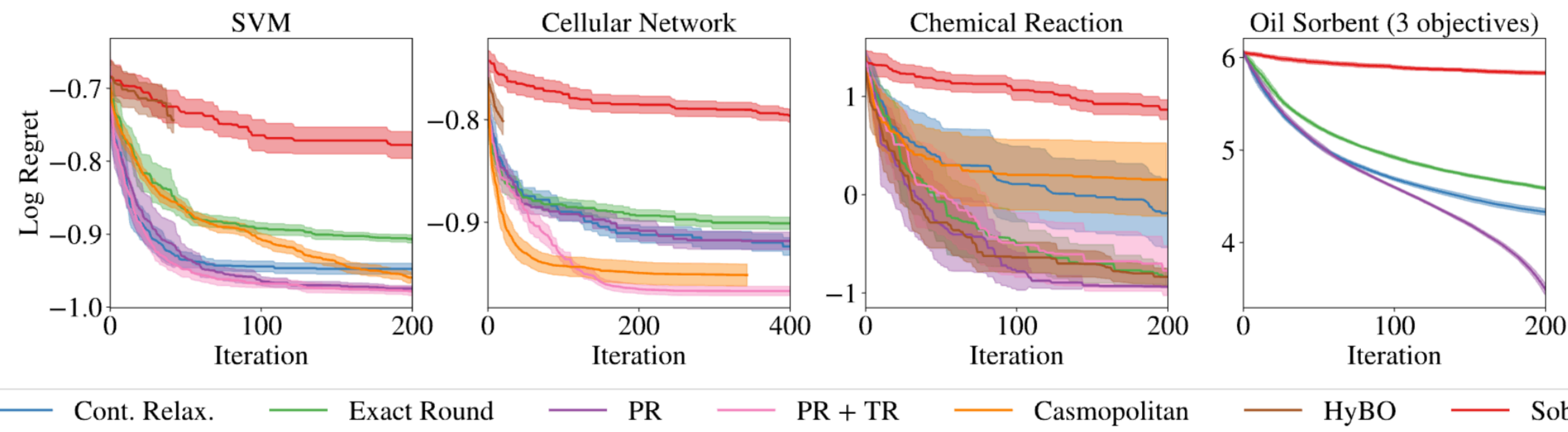
Improved support
in Ax & BoTorch
currently in the
works!

- Many optimization problems include discrete (ordered or categorical) parameters: $\mathbb{X} = \mathcal{X} \times \mathcal{Z}$
- Ax and BoTorch support discrete parameters out of the box
 - However, common approaches (such as one-hot encoding or continuous relaxation) do not scale well and have other issues
- New approach: **Probabilistic Reparameterization**
 - Key idea: Instead of optimizing over discrete Z , **optimize over probability distribution** $Z \sim p(Z | \theta)$, with θ a (continuous) parameter
 - Optimizer of reparameterization recovers optimizer of discrete problem.

Algorithm 1 BO with PR

```

1: Input: black-box objective  $f : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbb{R}$ 
2: Initialize  $\mathcal{D}_0 \leftarrow \emptyset$ ,  $\text{GP}_0 \leftarrow \text{GP}(\mathbf{0}, k)$ 
3: for  $n = 1$  to  $N_{\text{iterations}}$  do
4:    $(\mathbf{x}_n, \boldsymbol{\theta}_n) \leftarrow \arg \max_{(\mathbf{x}, \boldsymbol{\theta}) \in \mathcal{X} \times \Theta} \mathbb{E}_{\mathbf{Z} \sim p(\mathbf{Z} | \boldsymbol{\theta})} [\alpha(\mathbf{x}, \mathbf{Z})]$ 
5:   Sample  $\mathbf{z}_n \sim p(\mathbf{Z} | \boldsymbol{\theta}_n)$ 
6:   Evaluate  $f(\mathbf{x}_n, \mathbf{z}_n)$ 
7:    $\mathcal{D}_n \leftarrow \mathcal{D}_{n-1} \cup \{(\mathbf{x}_n, \mathbf{z}_n, f(\mathbf{x}_n, \mathbf{z}_n))\}$ 
8:   Update posterior  $\text{GP}_n$  given  $\mathcal{D}_n$ 
9: end for
  
```



Resources

- Website (general docs, tutorials, API docs):
 - ax.dev
 - botch.org
- GitHub (issues, discussions, latest features):
 - <https://github.com/facebook/Ax>
 - <https://github.com/pytorch/botorch>
- PyTorch Ax Multi-Objective Neural Architecture Search tutorial:
 - https://pytorch.org/tutorials/intermediate/ax_multiobjective_nas_tutorial.html

PRs welcome



Ax



BoTorch