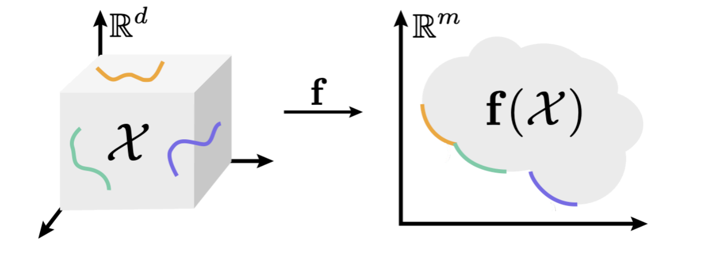
Diversity-Guided Multi-Objective Bayesian Optimization With Batch Evaluations

Mina Konaković Luković*, Yunsheng Tian*, Wojciech Matusik
MIT CSAIL
{mina, yunsheng, wojciech}@csail.mit.edu

: https://github.com/yunshengtian/DGEMO

Problem Setup

Given a design optimization problem with d design parameters and m different conflicting objectives:



our goal is to find the optimal set of design samples

$$\mathbf{X}^* = \arg\min_{\mathbf{X}} (f_1(\mathbf{X}), \ldots, f_m(\mathbf{X}))$$

that minimizes the objectives with different trade-offs, called *Pareto set*. The set of optimal objective values is called *Pareto front*.

In many real-world science and engineering problems, the objective functions are:

- **Black-box**
- **Expensive to evaluate**
- **Can be evaluated in parallel (batch)**

Contributions

We propose a novel algorithm called **DGEMO** (Diversity-Guided Efficient Multi-objective Optimization) for solving multi-objective black-box optimization problems, which:

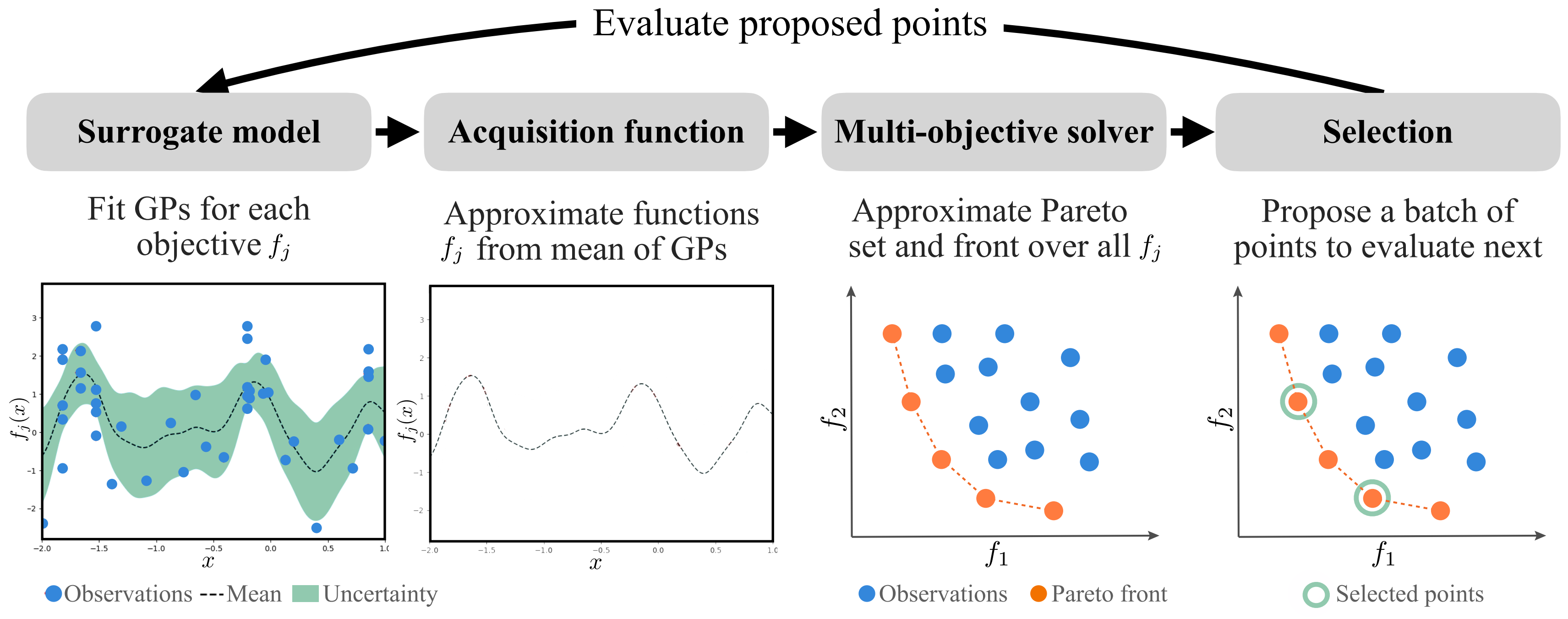
- Minimizes the number of function evaluations
- Enables batch evaluations with any batch sizes
- Achieves consistent and state-of-the-art performance on both synthetic test functions and real-world benchmark problems

Takeaways

If you have any design optimization problem that has multiple conflicting objectives and is expensive to evaluate, please try our DGEMO for state-of-the-art efficiency.

Algorithm Pipeline

Our algorithm follows an iterative multi-objective Bayesian optimization pipeline:



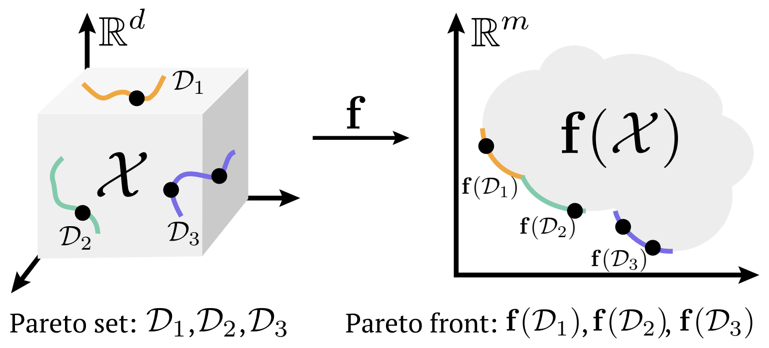
Repeat until a stopping criterion is met:

1) Fit Gaussian processes (GPs) for each objective based on the current dataset. GPs are the surrogate models that map from design input to performance output.
2) Take the mean functions of fitted GPs as acquisition functions, that serve as evaluation functions purely based on surrogate models.
3) Run our multi-objective solver[1] on the acquisition functions to approximate the Pareto set and front, which results in a dense set of candidate points for selection.
4) Select a batch of points to evaluate next with our diversity-guided selection strategy.
5) Evaluate the selected points in real experiments and add them to the dataset.

Diversity-Guided Selection

We use a two-stage selection strategy that trades-off exploration and exploitation:

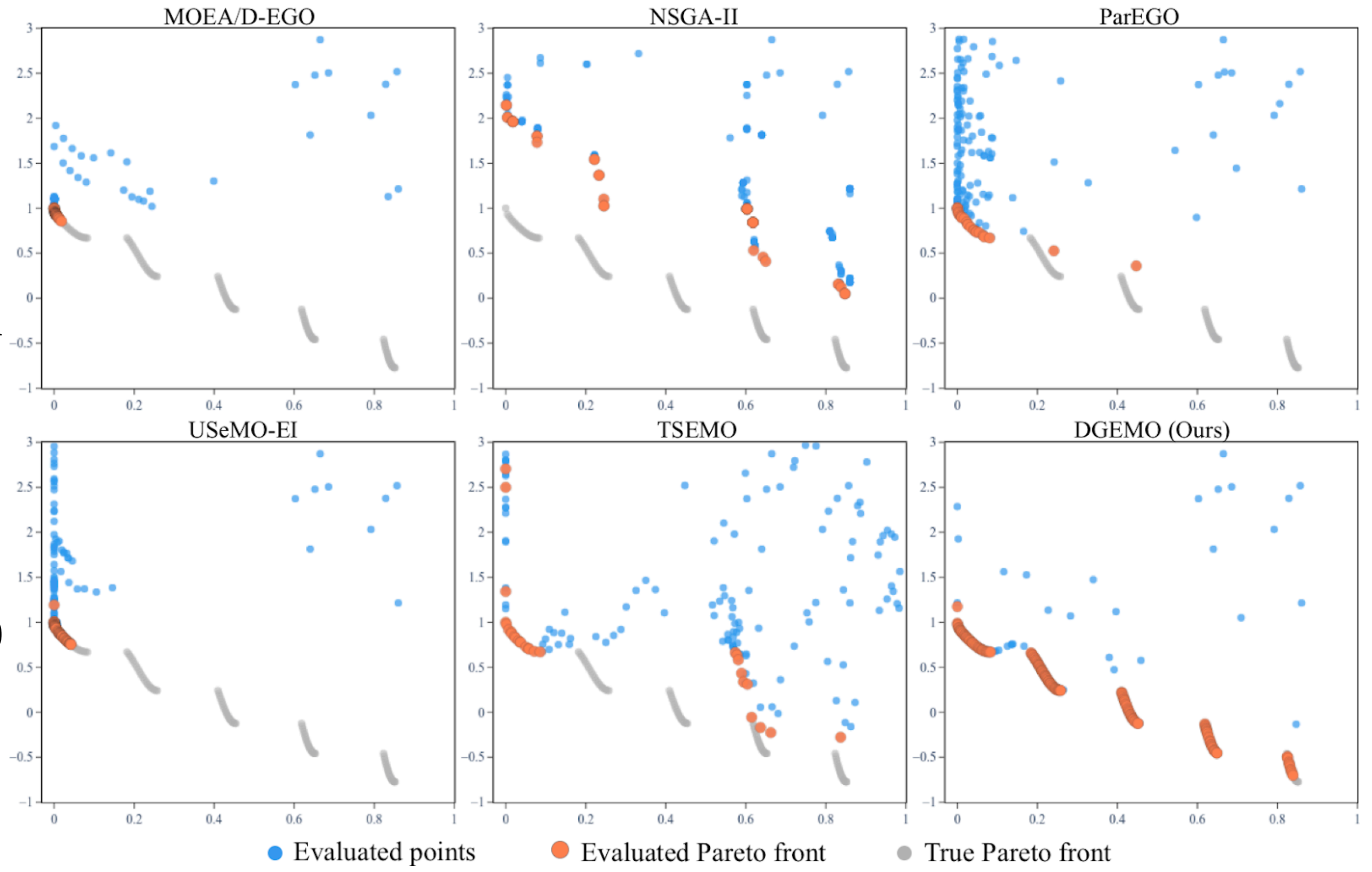
- **Diversity measure**: Considers diversity in both design and performance space of candidate optimal samples, and evenly distributes selected samples among *diversity regions* (given by our solver, shown in different colors). It encourages exploring potentially promising regions and prevents from falling into local minima.

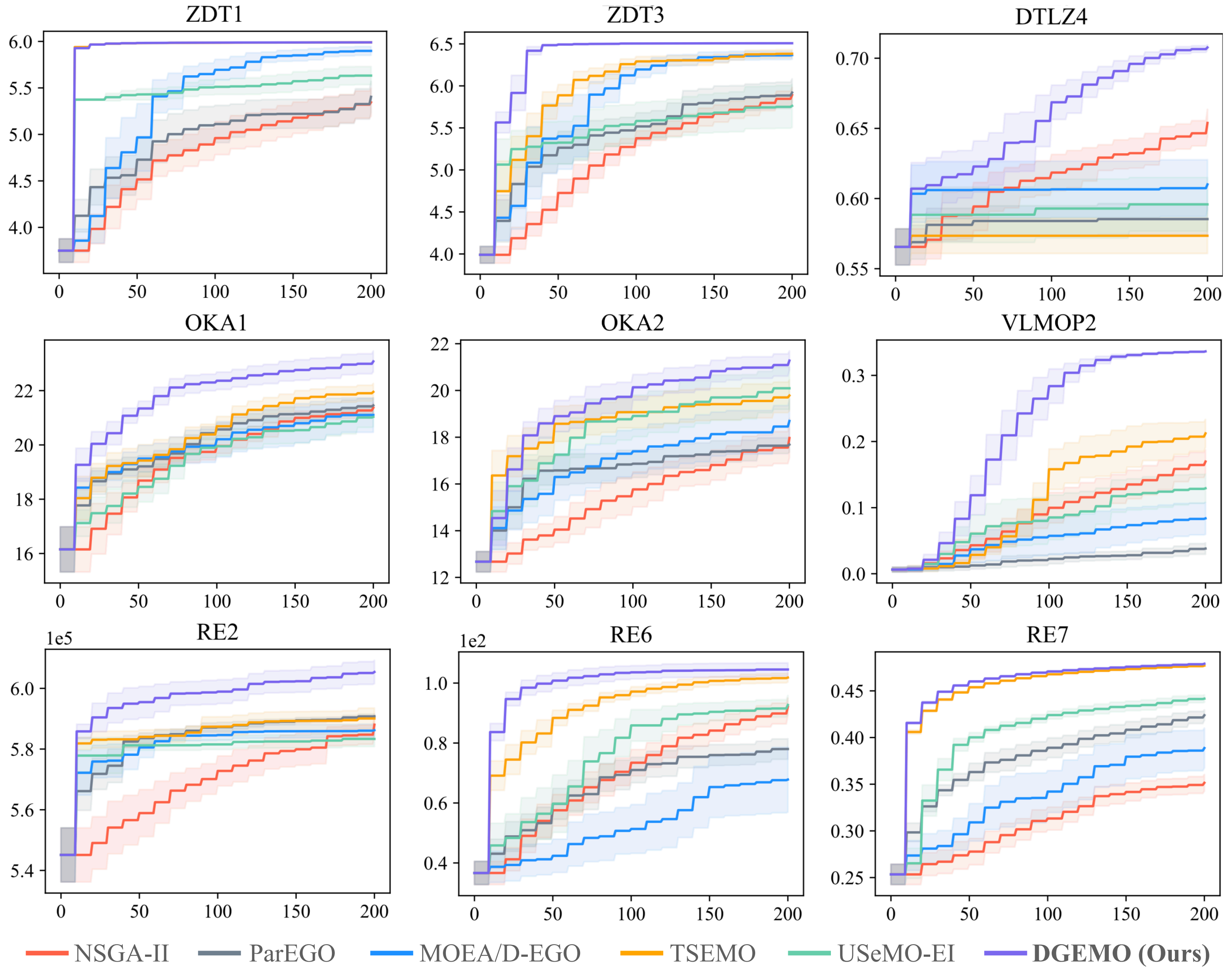


- **Hypervolume improvement**[2]: As the most popular performance criterion in multi-objective optimization, we use the hypervolume to rank points within a diversity region and select the ones with top ranks for experimental evaluation.

Experimental Results

Qualitative →
Take the standard ZDT3 problem as an example. We notice that compared to other baseline algorithms, DGEMO quickly discovers all regions of interest and focuses most evaluations in these regions, covering almost the entire ground truth Pareto front after only 20 iterations of batch size 10 without getting stuck in local minima (see the bottom right figure).





← Quantitative
We monitor the hypervolume improvement of all algorithms on 13 synthetic test functions and 7 real-world benchmark problems (here we show 9 of them), averaged over 10 runs using different random seeds. DGEMO shows consistent top performance, compared to other algorithms that oscillate on different types of problems. Please check out the paper for full results on all problems with different batch sizes.

Reference

[1] Adriana Schulz, Harrison Wang, Eitan Grinspun, Justin Solomon, and Wojciech Matusik. Interactive exploration of design trade-offs. ACM Transactions on Graphics (TOG), 37(4):1–14, 2018.
[2] Nery Riquelme, Christian Von Lücken, and Benjamin Baran. Performance metrics in multiobjective optimization. In 2015 Latin American Computing Conference (CLEI), pages 1–11. IEEE, 2015.