RoMA: Robust Model Adaptation for Offline Model-Based Optimization

Sihyun Yu (KAIST), Sungsoo Ahn (MBZUAI), Le Song (MBZUAI, Biomap), Jinwoo Shin (KAIST)

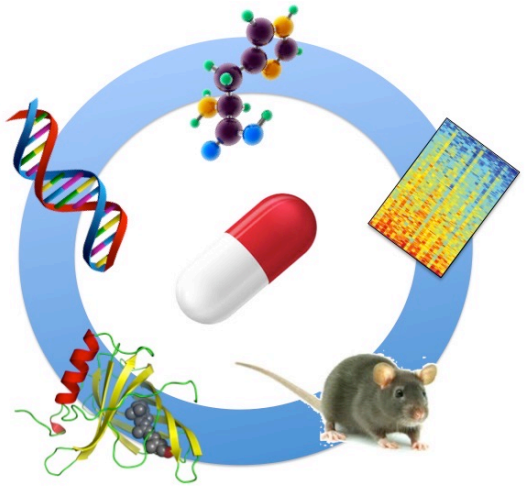
Introduction: Design Problem

Designing new objects with desired property is a fundamental problem in various domains.

- Including biology, chemistry, robotics, aircraft design.

Main Goal: find a new input $x^* \in \mathbb{R}^L$ that maximizes the black-box function $f^* : \mathbb{R}^L \to \mathbb{R}$

- Which is typically expensive to be evaluated.



$$x^* = \arg\max_{x \in \mathbb{R}^L} f^*(x)$$

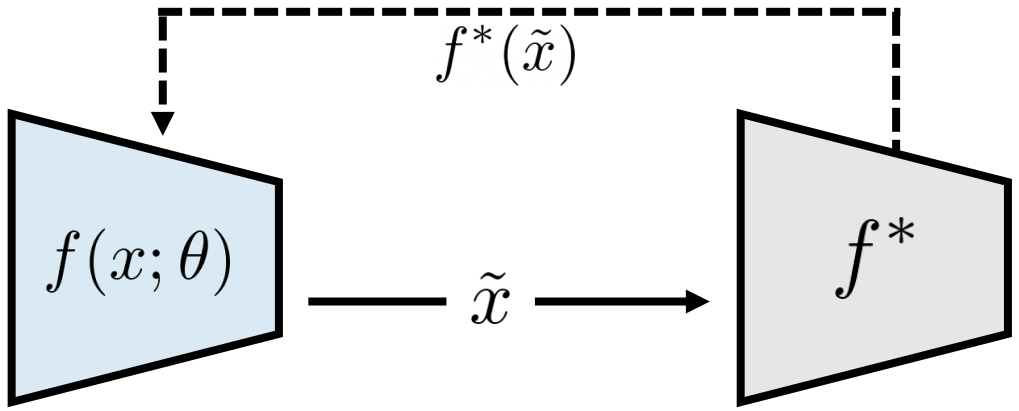
Introduction: Model-based Optimization

Challenge: Black-box objective function $f^*(x)$

- Moreover, evaluation often accompanies expensive costs (e.g., protein synthesis).

Common Approach: Model-based optimization

- Use a cheap proxy model $f(x;\theta)$ which approximates $f^*(x)$, i.e., $f(x;\theta) \approx f^*(x)$.
- After that, find a surrogate solution $\tilde{x}$ which maximizes the proxy model: $\tilde{x} = \arg\max_{x \in \mathbb{R}^L} f(x;\theta)$
- **Online MBO:** access to the black box function $f^*(x)$ is possible



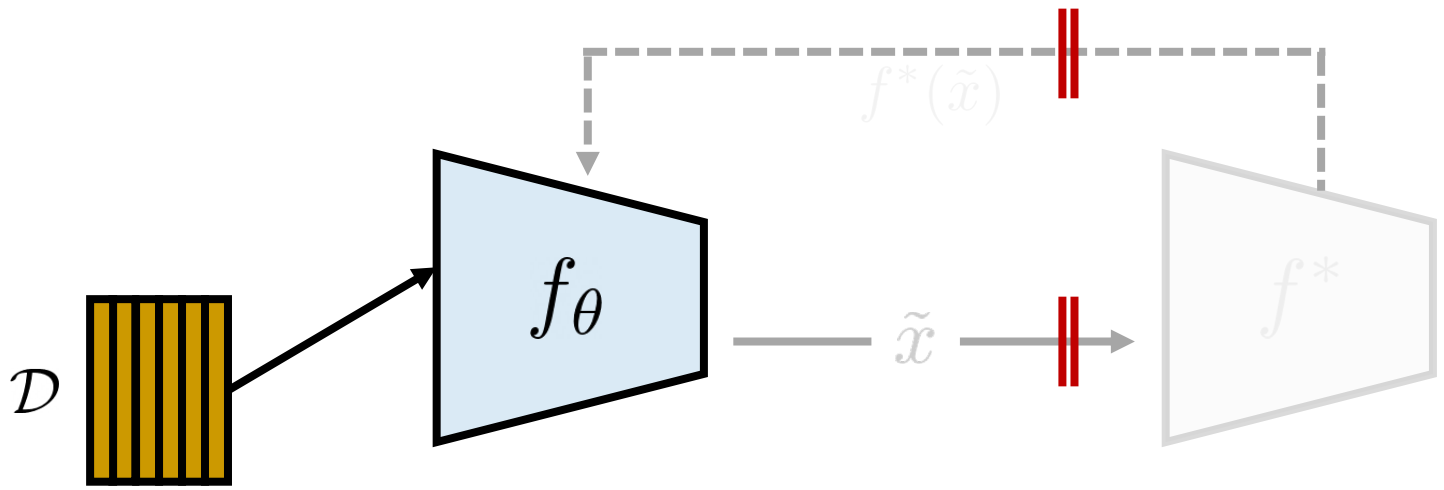
Introduction: Offline Model-based Optimization

Recall: New evaluation of black-box function f^* is mostly difficult.

- It often contains serious danger or expensive cost for evaluation. (e.g., aircraft design)

Recent Approach: 'Offline' model-based optimization (Offline MBO)

- Only offline dataset from previous observations is allowed: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{d}$
- No additional function queries are allowed: no access to f^* with the new inputs.

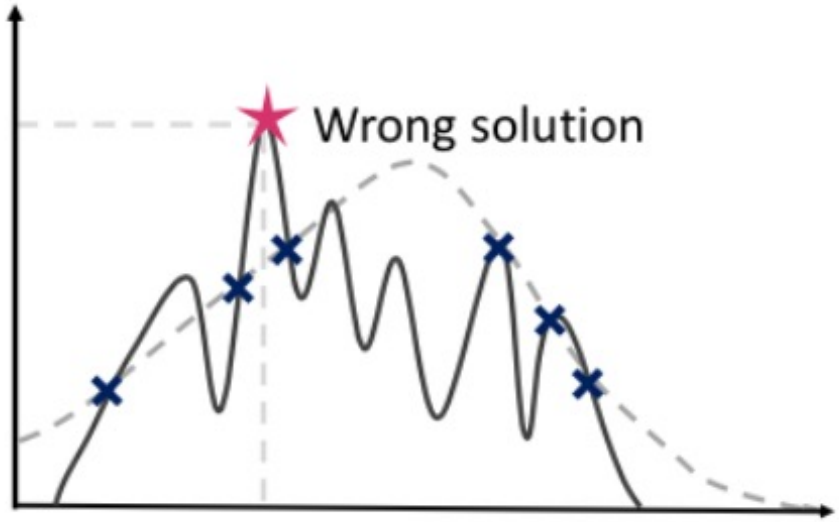


Goal: Offline Model-based Optimization

Goal: Generalizable proxy model outside the training data.

Challenge: Unexpected output from the learned DNN proxy model $f(x;\theta)$ [Fu et al., 2021, Trabucco et al., 2021]

- Why?: Overfitting at the training data: too sharp minima;
- May leads to wrong solution



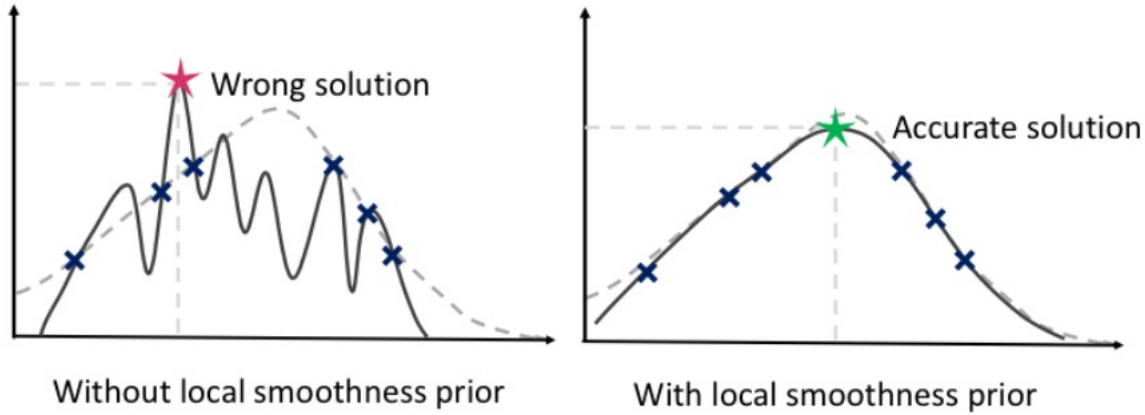
[Fu et al., 2020] Offine Model-based Optimization via Normalized Maximum Likelihood, ICLR 2021.
[Trabucco et al., 2021] Conservative Objective Models: A Simple Approach to Effective Model-based Optimization, ICML 2021

Our Idea: Regularizing Smoothness Prior

Goal: Generalizable proxy model outside the training data.

Question: What is a effective prior for **regularizing the proxy model** $f(x;\theta)$ **?**

Idea: We propose to utilize **regularizations based on the** smoothness prior!



Method 1: Regularization for general data points (Robust Pre-training)

Method 2: Regularization for the current solution candidate (Model Adaptation)

Motivation: Smoothness Prior

Smoothness prior: Have known to enhance the generalization in various situations [Rosca et al., 2020]

- Weight decay [Ilya et al., 2018]
- Spectral regularization [Yuichi et al., 2018]
- Gradient norm penalty [Jure et al., 2017; Michael et al., 2018]

Moreover, they are highly correlated to adversarial robustness; (= smooth at the worst direction)

- Shows empirical correlation: [Novak et al., 2018, Szegedy et al., 2013]
- Theoretical justification [Justin et al., 2018]

[Rosca et al., 2020] A Case for New Neural Network Smoothness Constraints, NeurIPS 2020W.
[Ilya et al., 2018] Decoupled Weight Decay Regularization, ICLR 2018
[Yuichi et al., 2018] Spectral Norm Regularization for Improving the Generalization, ICLR 2018
[Jure et al., 2017] Robust Large Margin Deep Neural Networks, IEEE Transactions on Signal Processing, 2017
[Michael Arbel, 2018] On Gradient Regularizers for MMD GANs, NeurIPS 2018
[Novak et al., 2018] Sensitivity and Generalization
[Szegedy et al., 2013] Intriguing properties of Neural Networks, ICLR 2013
[Justin et al., 2018] Adversarial Spheres, 2018

Overview: Robust Model Adaptation (RoMA)

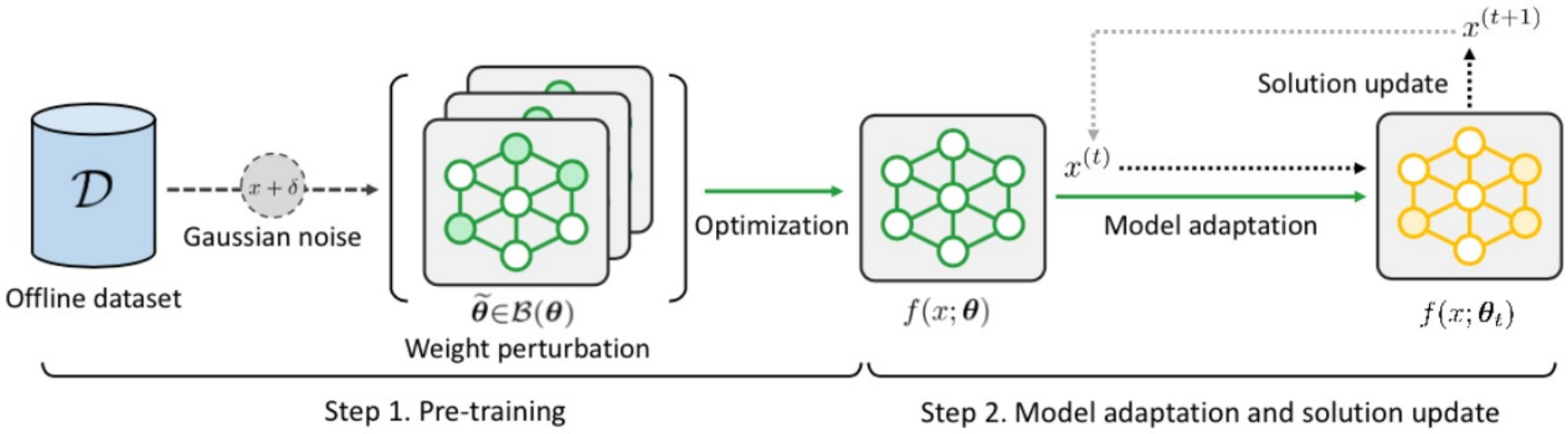
RoMA: We propose a novel offline MBO framework to adaptively adjust the model.

- Consists of two stage procedure.

At a high level, RoMA is operated as follows:

$$\theta_{-1} = \arg\min \mathcal{L}\,, x^{(0)} \in \mathcal{D}$$

$$\begin{aligned}
&\texttt{for } t = 0 \texttt{ to } T \\
&\qquad \theta_t = \texttt{update}(x^{(t)}, \theta_{t-1}, \theta_{-1}) \\
&\qquad x^{(t+1)} = \texttt{gradient-update}(x^{(t)}, \theta_t)
\end{aligned}$$



Overview: Robust Model Adaptation (RoMA)

RoMA: We propose a novel offline MBO framework to adaptively adjust the model.

- Consists of **two stage procedure**

Stage 1. Robust pre-training of the proxy model.

- **Method 1:** Regularization for general data points

$$\text{Minimize} \max_{\widetilde{\theta} \in \mathcal{B}(\theta)} \left[\mathbb{E}_{(x,y)\sim\mathcal{D}} \left[(f(x;\widetilde{\theta}) - y)^2 \right] \right] \text{ over } \theta$$

$$\text{where } \mathcal{B}(\theta) := \left\{ \tilde{\theta} : \|\theta_\ell - \tilde{\theta}_\ell\|_F \leq \epsilon \cdot \|\theta_\ell\|_F \quad \forall\ \ell = 1, \cdots, L \right\}.$$

Paramter of ℓ-th layer matrix

Stage 2. Model adaptation & gradient-based solution update

- **Method 2:** Regularization for the current solution candidate

$$\theta_t = \arg\min_{\tilde{\theta} \in \mathcal{B}(\theta)} \left[\|\nabla_x f(x;\tilde{\theta})\|_2 \Big|_{x=x^{(t)}} + \alpha \left(f(x^{(t)};\tilde{\theta}) - f(x^{(t)};\theta_{t-1}) \right)^2 \right], \quad \theta_{-1} = \theta$$

$$x^{(t+1)} := x^{(t)} + \eta\ \nabla_x f(x;\theta_t)|_{x=x^{(t)}}$$

Stage 1. Robust Pre-training

Stage 1. Robust pre-training of the proxy model.

- Worst-case optimization to the weight perturbation.
- Motivated by the regularization proposed in [Wu et al., 2020]

$$\text{Minimize} \max_{\widetilde{\theta} \in \mathcal{B}(\theta)} \left[\mathbb{E}_{(x,y)\sim\mathcal{D}} \left[(f(x;\widetilde{\theta}) - y)^2 \right] \right] \text{ over } \theta$$

$$\text{where } \mathcal{B}(\theta) := \left\{ \tilde{\theta} : \|\theta_\ell - \tilde{\theta}_\ell\|_F \leq \epsilon \cdot \|\theta_\ell\|_F \quad \forall\ \ell = 1, \cdots, L \right\}.$$

Paramter of ℓ-th layer matrix

Note: We are utilizing Gaussian noise **data augmentation** while training.

- For input-level smoothness regularization.

[Cohen et al., 2019] Carried Adversarial Robustness via Randomized Smoothing, ICML 2019

Stage 2. Model Adaptation & Solution Update

Stage 2. Model adaptation & Solution Update

- We update the solution at **the adjusted model.**

$$\theta_t = \arg\min_{\tilde{\theta} \in \mathcal{B}(\theta)} \Big[\, ||\nabla_x f(x;\tilde{\theta})||_2 \Big|_{x=x^{(t)}} + \alpha \big(f(x^{(t)};\tilde{\theta}) - f(x^{(t)};\theta_{t-1}) \big)^2 \Big], \quad \theta_{-1} = \theta$$

$$x^{(t+1)} := x^{(t)} + \eta \, \nabla_x f(x;\theta_t)|_{x=x^{(t)}}$$

Note: Adaptation leads the update at the model which satisfies:

- To maintain accurate prediction at the training dataset
- Smooth at the current solution $x^{(t)}$

Stage 2. Model Adaptation & Solution Update

Stage 2. Model adaptation & Solution Update

- We update the solution at **the adjusted model.**

$$\theta_t = \arg\min_{\tilde{\theta}\in\mathcal{B}(\theta)} \Big[||\nabla_x f(x;\tilde{\theta})||_2 \Big|_{x=x^{(t)}} + \alpha\big(f(x^{(t)};\tilde{\theta}) - f(x^{(t)};\theta_{t-1})\big)^2 \Big], \quad \theta_{-1} = \theta$$

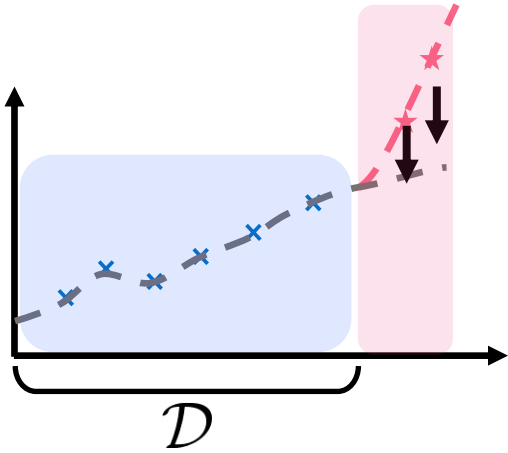
$$x^{(t+1)} := x^{(t)} + \eta\, \nabla_x f(x;\theta_t)|_{x=x^{(t)}}$$

Note: Adaptation leads the update at the model which satisfies:

- To maintain accurate prediction at the training dataset
- Smooth at the current solution $x^{(t)}$

Remark 1: Why adaptive framework?

- Iterative update via gradient ascent – causes distributional shift; **requires adjustment**



Stage 2. Model Adaptation & Solution Update

Stage 2. Model adaptation & Solution Update

- We update the solution at **the adjusted model.**

$$\theta_t = \arg\min_{\tilde{\theta} \in \mathcal{B}(\theta)} \Big[||\nabla_x f(x;\tilde{\theta})||_2 \Big|_{x=x^{(t)}} + \alpha \big(f(x^{(t)};\tilde{\theta}) - f(x^{(t)};\theta_{t-1}) \big)^2 \Big], \quad \theta_{-1} = \theta$$

$$x^{(t+1)} := x^{(t)} + \eta \, \nabla_x f(x;\theta_t)|_{x=x^{(t)}}$$

Note: Adaptation leads the update at the model which satisfies:

- To maintain accurate prediction at the training dataset
- Smooth at the current solution $x^{(t)}$

Remark 2: Minimizing a gradient norm

- Straightforward regularization; for make the model smooth at $x^{(t)}$

Stage 2. Model Adaptation & Solution Update

Stage 2. Model adaptation & Solution Update

- We update the solution at **the adjusted model.**

$$\theta_t = \arg\min_{\tilde{\theta} \in \mathcal{B}(\theta)} \Big[||\nabla_x f(x; \tilde{\theta})||_2 \Big|_{x=x^{(t)}} + \alpha \big(f(x^{(t)}; \tilde{\theta}) - f(x^{(t)}; \theta_{t-1}) \big)^2 \Big], \quad \theta_{-1} = \theta$$

$$x^{(t+1)} := x^{(t)} + \eta \, \nabla_x f(x; \theta_t)|_{x=x^{(t)}}$$

Remark 3: Utilize pseudo-label as the regularization

- **Note:** Repeating updates via gradient ascent leads distributional shift.
- Can be viewed as test time adaptation in image classification [Wang et al., 2021]

[Wang et al., 2019] Fully test-time adaptation by entropy minimization, ICLR 2021

Stage 2. Model Adaptation & Solution Update

Stage 2. Model adaptation & Solution Update

- We update the solution at **the adjusted model.**

$$\theta_t = \arg\min_{\tilde{\theta}\in\mathcal{B}(\theta)} \Big[||\nabla_x f(x;\tilde{\theta})||_2 \Big|_{x=x^{(t)}} + \alpha\big(f(x^{(t)};\tilde{\theta}) - f(x^{(t)};\theta_{t-1})\big)^2 \Big], \quad \theta_{-1} = \theta$$

$$x^{(t+1)} := x^{(t)} + \eta\, \nabla_x f(x;\theta_t)|_{x=x^{(t)}}$$

Remark 4: Accurate prediction at the dataset $\mathcal{D}$

- We have flat loss landscape to the model parameter; achieved by robust pre-training

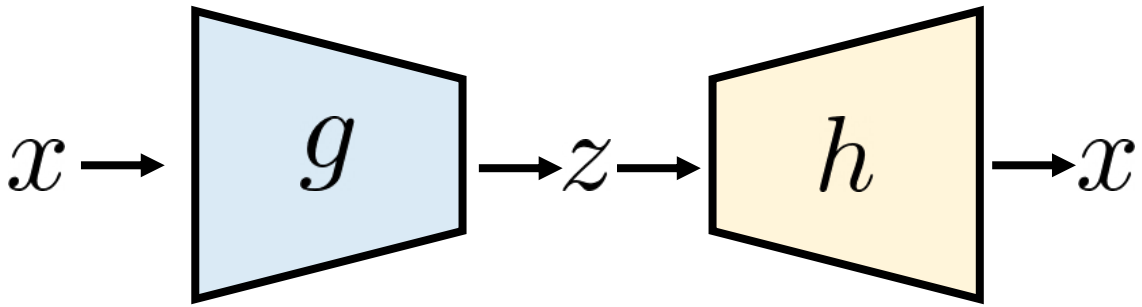
Remark 5: Consider only $x^{(t)}$ for adjustment? No other inputs?

- As we are updating the solution via gradient ascent (which is **'local update'**)

Idea: Handling Discrete Inputs

Recall: Discrete inputs + Gradient ascent is problematic

Idea: Utilize VAE [Kingma et al., 2014] and perform optimization on the latent space



$$f(x;\theta) := \tilde{f}(g(x);\theta) \approx f^*(x)$$

$$\tilde{z} = \arg\max_z \tilde{f}(z;\theta)$$

$$\tilde{x} = h(\tilde{z})$$

[Kingma et al., 2014] Auto-Encoding Variational Bayes, ICLR 2014

Experiments: Main Result (100th)

Verified on offline model-based optimization benchmark, Design-bench [Trabucco et al., 2021]

We start from initial 128 high-scored inputs from the dataset.

- Averaged over 16 runs
- 100th: best score / 50th: median score

Table 1: Comparison of 100th percentile scores for each task. We mark the scores within one standard deviation from the highest average score to be bold.

	Discrete domain		Continuous domain				
Method	GFP	Molecule	Supercond.	Hopper	Ant	Dkitty	Avg.†
Dataset Max	3.152	6.558	73.90	1361.6	108.5	215.9	1.000
CbAS [5]	**3.408±0.029**	6.301±0.131	72.17±8.652	547.1±423.9	393.0±3.750	**396.1±60.65**	1.324
Autofocus [8]	3.365±0.023	6.345±0.141	77.07±11.11	443.8±142.9	386.9±10.58	376.3±47.47	1.286
NEMO [10]	3.359±0.036	6.682±0.209	**127.0±7.292**	2130.1±506.9	393.7±6.135	**431.6±47.79**	1.687
MINs [24]	3.315±0.029	6.508±0.236	80.23±10.67	746.1±636.8	388.5±9.085	352.9±38.65	1.304
COMs [51]	3.305±0.024	**6.876±0.128**	110.0±6.804	**2395.7±561.7**	378.8±10.01	341.4±28.47	1.589
Grad. Ascent [52]	2.894±0.001	6.636±0.066	89.64±9.201	1050.8±284.5	399.9±4.941	**390.7±49.24**	1.237
RoMA (Ours)	3.357±0.024	**6.890±0.122**	103.9±5.487	**2466.5±359.2**	**468.5±12.68**	**384.3±51.68**	**1.705**

$$\frac{f^*(x) - y_{\min}}{y_{\max} - y_{\min}}$$

[Trabucco et al., 2021] Benchmarks for Data-Driven Offline Model-Based Optimization

Experiments: Main Result (50th)

Verified on offline model-based optimization benchmark, Design-bench [Trabucco et al., 2021]

We start from initial 128 high-scored inputs from the dataset.

- Averaged over 16 runs
- 100th: best score / 50th: median score

Table 2: Comparison of 50th percentile scores for each task. We mark the scores within one standard deviation from the highest average score to be bold.

	Discrete domain		Continuous domain				
Method	GFP	Molecule	Supercond.	Hopper	Ant	Dkitty	Avg.†
Dataset Max	3.152	6.558	73.90	1361.6	108.5	215.9	1.000
CbAS [5]	**3.269±0.018**	5.472±0.123	32.21±7.255	132.5±23.88	267.3±16.55	203.2±3.580	0.826
Autofocus [8]	3.216±0.029	5.759±0.158	31.57±7.457	116.4±18.66	176.7±59.94	199.3±8.909	0.752
NEMO [10]	3.219±0.039	5.814±0.092	**66.41±4.618**	390.2±43.37	326.9±5.229	180.8±34.94	0.960
MINs [24]	3.135±0.019	5.806±0.078	37.32±10.50	520.4±301.5	184.8±29.52	211.6±13.67	0.803
Grad. Ascent [52]	2.894±0.000	**6.401±0.186**	54.06±5.06	185.0±72.88	318.0±12.05	**255.3±6.379**	0.862
RoMA (Ours)	3.230±0.015	6.160±0.018	**68.99±6.687**	**560.1±22.47**	**370.8±6.771**	**252.4±5.167**	**1.103**

$$\frac{f^*(x) - y_{\min}}{y_{\max} - y_{\min}}$$

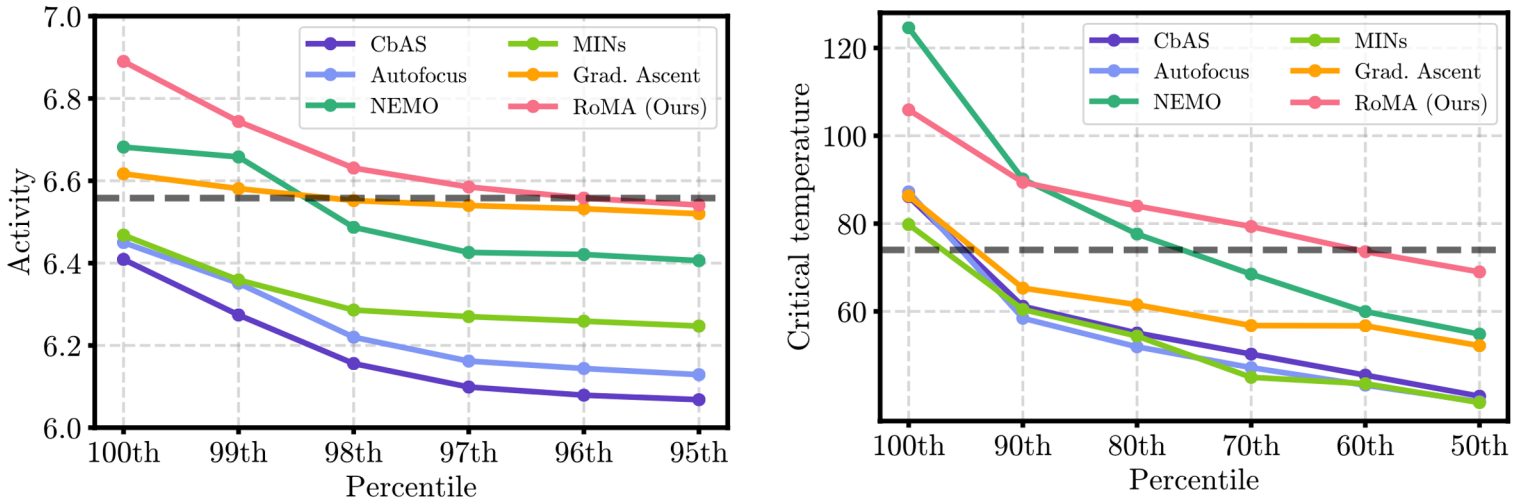
[Trabucco et al., 2021] Benchmarks for Data-Driven Offline Model-Based Optimization

Ablation Study: Ratio of high-scoring solutions

Note: maximum score in the dataset can be a naïve baseline of offline MBO

Question: Is RoMA beneficial at having more high-scoring solutions than the best offline data?

- RoMA shows its superiority in this perspective.



(a) Molecule

(b) Superconductor

Figure 3: Scores of the ground-truth objective function evaluated at samples of different percentiles in Molecule and Superconductor task. The dotted lines indicate the maximum score in the dataset.

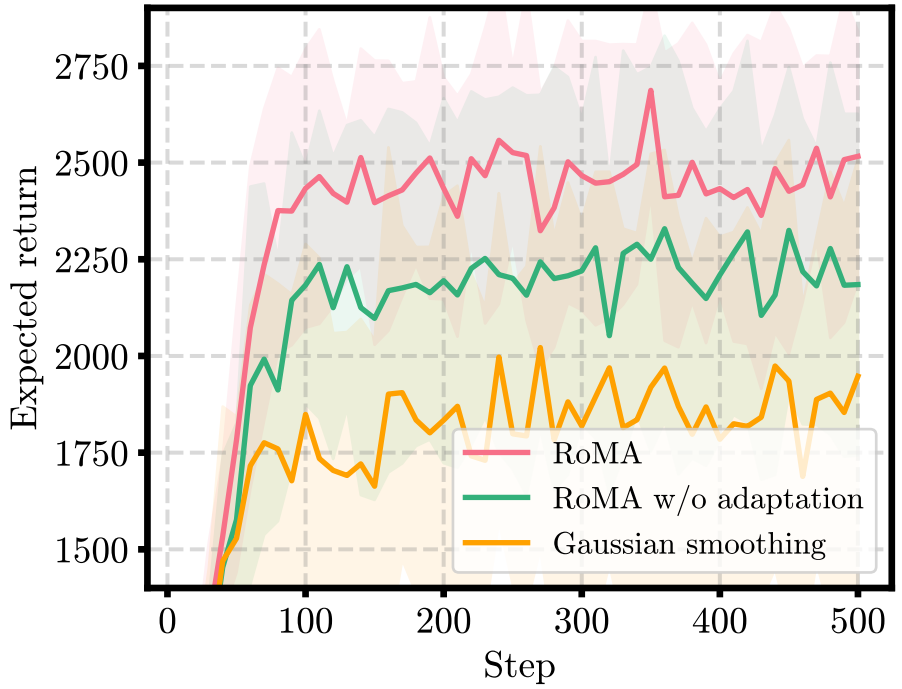
Ablation Study: Effect of Each Component

Question 1. Does employing '**robust pre-training**' is helpful?

- Robust pre-training itself is certainly beneficial.

Question 2. Does '**model adaptation**' further shows more improvement?

- Confirms the orthogonal improvement of model adaptation.



Conclusion & Discussion

RoMA: We propose a novel offline MBO framework to adaptively adjust the model.

- We use "smoothness prior" to regularize the proxy model for better generalization

RoMA consists of two-stage procedure.

- Stage 1. Robust pre-training of the proxy model.
- Stage 2. Model adaptation & Gradient-based solution update

RoMA is beneficial at various offline MBO tasks.

- Outperform all at 4,4 tasks at 100^{th}, 50^{th} score, respectively.
- State-of-the-art in average.