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 \( f^* : \mathbb{R}^L \rightarrow \mathbb{R} \)
- 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
Recall: New evaluation of black-box function $f^\ast$ 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 $f^\ast$ 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

[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 an 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]
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:

\[
\theta_{-1} = \arg \min_{\theta} \mathcal{L}, x^{(0)} \in \mathcal{D}
\]

for \( t = 0 \) to \( T \)

\[
\theta_t = \text{update}(x^{(t)}, \theta_{t-1}, \theta_{-1})
\]

\[
x^{(t+1)} = \text{gradient-update}(x^{(t)}, \theta_t)
\]
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

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

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

Stage 2. Model adaptation & gradient-based solution update

- Method 2: Regularization for the current solution candidate

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

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

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

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

**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

- We update the solution at the adjusted model.

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

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

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.

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

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

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.

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

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

**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.

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

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

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

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

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

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

**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
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

Experiments: Main Result (100th)

<table>
<thead>
<tr>
<th>Method</th>
<th>Discrete domain</th>
<th>Continuous domain</th>
<th>Avg.†</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset Max</td>
<td>3.152</td>
<td>6.558</td>
<td>1.000</td>
</tr>
<tr>
<td>CbAS [5]</td>
<td>3.408±0.029</td>
<td>6.301±0.131</td>
<td>396.1±60.65</td>
</tr>
<tr>
<td>Autofocus [8]</td>
<td>3.365±0.023</td>
<td>6.345±0.141</td>
<td>376.3±47.47</td>
</tr>
<tr>
<td>NEMO [10]</td>
<td>3.359±0.036</td>
<td>6.682±0.209</td>
<td>431.6±47.79</td>
</tr>
<tr>
<td>MINs [24]</td>
<td>3.315±0.029</td>
<td>6.508±0.236</td>
<td>352.9±38.65</td>
</tr>
<tr>
<td>COMs [51]</td>
<td>3.305±0.024</td>
<td>6.876±0.128</td>
<td>341.4±28.47</td>
</tr>
<tr>
<td>Grad. Ascent [52]</td>
<td>2.894±0.001</td>
<td>6.636±0.066</td>
<td>390.7±49.24</td>
</tr>
<tr>
<td>RoMA (Ours)</td>
<td>3.357±0.024</td>
<td>6.890±0.122</td>
<td>384.3±51.68</td>
</tr>
</tbody>
</table>

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.

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

[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>
<thead>
<tr>
<th>Method</th>
<th>Discrete domain</th>
<th>Continuous domain</th>
<th>Avg.†</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GFP</td>
<td>Molecule</td>
<td></td>
</tr>
<tr>
<td>Dataset Max</td>
<td>3.152</td>
<td>6.558</td>
<td></td>
</tr>
<tr>
<td>CbAS [5]</td>
<td>3.269±0.018</td>
<td>5.472±0.123</td>
<td>0.826</td>
</tr>
<tr>
<td>Autofocus [8]</td>
<td>3.216±0.029</td>
<td>5.759±0.158</td>
<td>0.752</td>
</tr>
<tr>
<td>NEMO [10]</td>
<td>3.219±0.039</td>
<td>5.814±0.092</td>
<td>0.960</td>
</tr>
<tr>
<td>MINs [24]</td>
<td>3.135±0.019</td>
<td>5.806±0.078</td>
<td>0.803</td>
</tr>
<tr>
<td>Grad. Ascent [52]</td>
<td>2.894±0.000</td>
<td>6.401±0.186</td>
<td></td>
</tr>
<tr>
<td>RoMA (Ours)</td>
<td>3.230±0.015</td>
<td>6.160±0.018</td>
<td>1.103</td>
</tr>
</tbody>
</table>

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

\[
f^*(x) - y_{\text{min}} \quad \frac{y_{\text{max}} - y_{\text{min}}}{y_{\text{max}} - y_{\text{min}}}
\]
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.

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 100th, 50th score, respectively.
• State-of-the-art in average.