Constrained Offline Black-Box Optimization via Risk Evaluation and Management

Yiyi Zhu[1], Huakang Lu[1], Yupeng Wu[1], Shuo Liu[1], Jing-Wen Yang[2], Hong Qian[1]*

[1]School of Computer Science and Technology, East China Normal University, Shanghai 200062, China
[2]Intelligent NPC Team, Game AI Center, Tencent Inc, Shenzhen 518057, China
{yiyizhu, huakang.lu, 51215901031, shuoliu}@stu.ecnu.edu.cn, jingwenyang@tencent.com, hqian@cs.ecnu.edu.cn

Abstract

Offline black-box optimization aims to identify the optimal solution of a black-box objective function under the guidance of a surrogate model constructed solely from a pre-collected dataset. It is commonly used in industrial scenarios, which often involve constraints, i.e., constrained offline optimization (COO). Offline optimization has progressed in addressing the out-of-distribution (OOD) issue caused by its inherent inability to interact with the objective function. However, there is not enough research in addressing more difficult scenarios, which must simultaneously address OOD issues and constrained issues to find stable, high-quality (i.e., high-scoring and feasible) solutions. To bridge this gap, this paper proposes a method called constrained offline optimization via risk evaluation and management (COOREM), which is capable of consistently surpassing the offline dataset under the condition of satisfying constraints. Specifically, COOREM employs a dual-energy model to separately evaluate OOD risk and constrained risk. This separation strategy aims to distinguish and address two difficult cases: the infeasible but not OOD solutions and the feasible but OOD solutions. Moreover, COOREM effectively manages OOD risk and constrained risk, ensuring the identification of high-quality solutions. Extensive experiments on real-world tasks, e.g., space missions, process synthesis, and design problems, showcase COOREM's effectiveness in managing both OOD risk and constrained risk. Furthermore, our findings indicate that COOREM could outperform online methods that need to access the objective function in certain space missions.

Introduction

Real-world design problems in numerous domains encounter widespread challenges spanning diverse fields such as materials science (Trabucco et al. 2022), robotics (Liao et al. 2019), credit risk assessment (Gu et al. 2024), and protein (Brookes, Park, and Listgarten 2019). Many methods generate new designs by optimizing an unknown objective function with active interaction to map designs to their property scores. Unfortunately, evaluating the objective function can be costly, risky, or even infeasible, such as synthesizing protein structures often depends on simulations, requiring extensive effort across multiple domains. Recently, researchers propose to access only prior data, i.e., offline datasets (Trabucco et al. 2022; Xue et al. 2024). They build a model via an offline dataset without access to the unknown objective function or any form of active interaction, and use the model to guide the search for suitable designs and evaluate. This can greatly reduce the budget for costly active interaction if properly utilized. This manner is known as *offline optimization*.

Offline optimization (Jin, Wang, and Sun 2021; Gong, Zhong, and Huang 2024) aims to find better (higher-scoring in this paper) designs than that in offline datasets by training a surrogate model to approximate the unknown objective function and performing optimization. Specifically, a common approach first trains a deep neural network (DNN) (Glorot, Bordes, and Bengio 2011) on the offline dataset to serve as a surrogate model. Optimization then begins from the optimal designs of the offline dataset, employing methods such as gradient ascent to identify the best possible designs in the unknown objective function. The best designs found by such a naïve method are usually far from the offline dataset. Regrettably, the regime of designs far from the offline dataset is often not well-trained, which results in out-of-distribution (OOD) issue, as mentioned by (Qi et al. 2022; Chen et al. 2022, 2023). For instance, as shown in Figure 1 (d), the blue dots represent the points that are overestimated due to the OOD issue. As the optimization proceeds, the optimization method might favor designs deemed high-scoring by the surrogate but perform poorly on the unknown objective function because of the gap between the surrogate and the unknown objective function.

Related Work. To address this issue, recent studies have proposed a range of approaches, which mainly include two broad classes: involving conservatism into the surrogate model or regularization in generative model. For the former, COMs (Trabucco et al. 2021) integrates a regularization term into the modeling of conservative objectives to depress adversarial designs. ROMA (Yu et al. 2021) overcomes the non-smooth nature of DNN by utilizing a local smoothness prior to achieving conservatism. IOM (Qi et al. 2022) enforces the surrogate to learn invariant representations and makes a conservative estimation. ARCOO (Lu et al. 2023) develops an energy model to quantify and manage OOD risk during optimization to obtain stable, high-scoring designs. For the latter, CbAs (Brookes, Park, and Listgarten 2019)

*Corresponding Author: hqian@cs.ecnu.edu.cn.

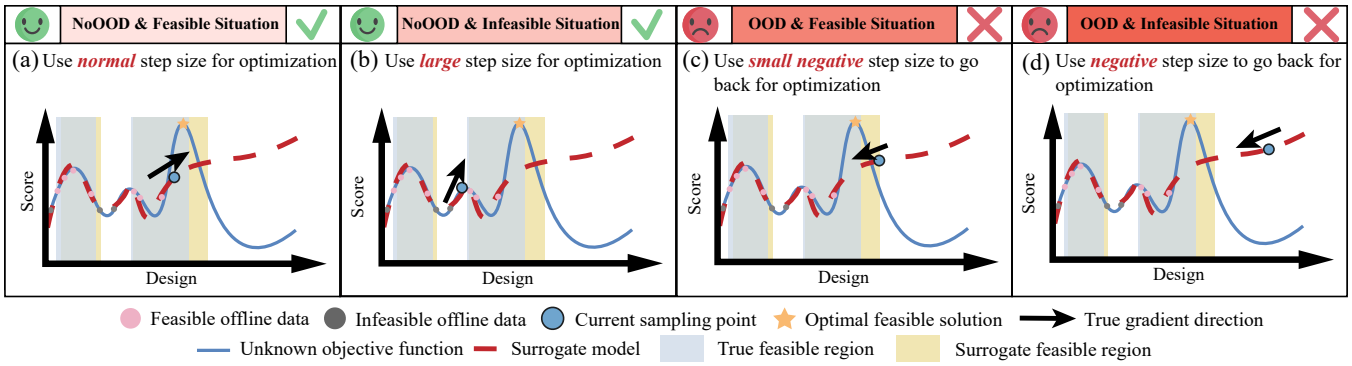


Figure 1: **Illustration of motivation**. The strategies of optimization are shown at four different optimization stages. The direction and length of the arrows in each subfigure respectively illustrate the optimization direction and the step size.

employs a variational auto-encoder (Kingma and Welling 2014) to characterize the distribution of designs and adopt the model to the optimal design in a trust region. MIN (Kumar and Levine 2020) learns a generator that maps the scores to the designs using a generative adversarial network (Goodfellow et al. 2014). Although prior work has been successful, it often depends on fixed-time optimization steps, where an algorithm's performance hinges on the termination criterion. In practice, offline optimization often uses anytime steps due to uncertain optimal step sizes for different tasks. Thus, a successful optimization process should consistently provide high-scoring designs regardless of when it terminates.

However, beyond the challenge of the unknown objective function and unachievable evaluations, many complex real-world scenarios are further complicated by constraints. This gives rise to *constrained offline optimization* (COO) problems. The goal of COO is to find high-quality (i.e., high-scoring and feasible) designs through a delicate symbiosis between the exploitation of the offline high-value, feasible designs and guided exploration to find designs that surpass the offline dataset, i.e., we need the surrogate to find out better designs than the offline dataset. As shown in Figure 1 (d), although we want to find out yellow stars, it may find the blue point due to OOD issues, i.e., the surrogate often makes inaccurate predictions and constrained estimations, complicating the symbiosis process. Consequently, successful COO methods inherently involve balancing the need to stay "near" the offline dataset to obtain accurate and feasible designs and the necessity to deviate from the offline dataset to find better designs. At the same time, a stable and feasible optimization process is also necessary in COO.

To address COO, current research primarily includes two ideas. One line of research with a series of methods (Huang and Wang 2021) is to handle constraints independently from the surrogate. This method, e.g., DE-PF, uses existing constraint-handling techniques to address constraints and integrates them with evolutionary algorithms to construct the surrogate model. However, a major limitation is that they can't effectively address the OOD issue in COO. On another front, PRIME (Kumar et al. 2022) involves assigning lower weights to infeasible designs to avoid these regions. However, the application of PRIME often requires elaborate tuning across different tasks. Though some promising methods have been proposed, there is a significant lack of research focused on the challenge that needs to achieve a balance between constrained satisfaction and performance improvement in the whole optimization process.

Motivation. COO aims to identify high-quality designs, yet differentiating the risk degrees between OOD and constraint remains challenging, often confounding the optimization process. Prior approaches often use smaller optimization steps under high-risk conditions (i.e., OOD or infeasible regions) or suppress these regions. However, they often fail to distinguish and address OOD and constrained risks, leading to suboptimal results. Quantifying and dynamically responding to both OOD and constrained risk is necessary, thereby facilitating more precise navigation through the design space. As shown in Figure 1, we should retreat in regions where both OOD and constraint risks are high, and proceed with a normal step size in areas where both risks are low. In cases where OOD risk is high but constraint risk is low, it is more prudent to back off because the constraint risk might be inaccurately estimated due to the elevated OOD risk. Conversely, when constraint risk is high but OOD risk is low, a larger step size should be chosen so as to quickly escape the infeasible region and search for a better design.

Contribution. Driven by this motivation, this paper proposes Constrained Offline Optimization via Risk Evaluation and Management (COOREM) method to find stable, high-quality designs under the COO scenario for real-world applications. COOREM introduces a three-stage model: adaptive surrogate model (ASM), composite risk evaluation model (CREM), and adaptive risk management (ARM) to control the risk adaptively in the optimization procedure. Specifically, CREM proposes a dual-energy model to express OOD risk and constrained risk. After that, we use contrastive divergence to train the risk, which aims to distinguish between high-risk (i.e., OOD or infeasible designs in the offline dataset) and low-risk (i.e., not OOD or feasible designs in the offline dataset). After risk is learned, ARM is used to manage the risk of the optimization process adaptively, which ensures that the risk is adjusted across various stages of the optimization process shown in Figure 1. Experimental results show that COOREM has significant advantages

in both key indicators of optimization stability and optimality. In addition, COOREM even outperforms online methods that need to access the objective function in some tasks.

In the following sections, we respectively introduce the problem definition, present the proposed COOREM, analyze the experimental results, and finally conclude the paper.

Problem Definition

Constrained Offline Optimization. Constrained offline optimization (COO) aims to find the optimum $\boldsymbol{x}^*$ which stably surpasses the offline dataset (i.e., the surrogate should find out better designs than the optimum in the offline dataset) through maximizing black-box objective function f:

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}\in\mathbb{R}^D} f(\boldsymbol{x}), \text{ s.t. } g_i(\boldsymbol{x}) \geq 0, i = 1, ..., n\,, \quad (1)$$

where $g_i(\boldsymbol{x})$ quantifies the constraint for designs, and n denotes the number of constraints. To achieve it, an offline dataset $\mathcal{D} = \mathcal{D}^+ \cup \mathcal{D}^- = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\} \cup \{\boldsymbol{x}'_1, \ldots, \boldsymbol{x}'_M\}$ is available, where $\mathcal{D}^+$ denotes the feasible dataset consisting of N feasible designs, and $\mathcal{D}^-$ denotes the infeasible dataset consisting of M infeasible designs. Among them, we consider designs $\boldsymbol{x}$ and the corresponding scores $f(\boldsymbol{x})$ which is applicable if and only if $\boldsymbol{x}$ is feasible.

A standard method for addressing the design problem involves fitting a DNN model to the offline dataset using supervised learning. The optimal parameters, $\boldsymbol{\theta}^*$, are determined by minimizing the mean squared error between the model's predictions and the actual scores: $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}(\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2$.

The trained DNN model, denoted as $\hat{f}_{\boldsymbol{\theta}}$, serves as a surrogate for optimizing the design through gradient ascent steps, i.e., $\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \eta\nabla_{\boldsymbol{x}}\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1})$, for $t = 1, \ldots, T$, where T denotes the number of steps, η represents the learning rate and $\boldsymbol{x}_T$ as the final design candidate.

Definition of COO's Risk. The first challenge is that its surrogate tends to produce inaccurate overestimations far from the offline dataset. Therefore, we define $\{\boldsymbol{x} \mid |\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x}) - f(\boldsymbol{x})| > \gamma\}$ as a set of the high-risk designs, which means there is a big gap γ between the surrogate $\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x})$ and the unknown function $f(\boldsymbol{x})$. The second challenge is that COO needs to deal with infeasible designs which we don't want to get. Therefore, we identify $\{\boldsymbol{x} \mid g(\boldsymbol{x}) < 0\}$ as another set of high-risk designs, i.e., $\boldsymbol{x}$ comes from the infeasible dataset $\mathcal{D}^-$. Moreover, designs are classified as low-risk when $\boldsymbol{x}$ comes from the feasible dataset $\mathcal{D}^+$. We regard the low-risk distribution $l(\boldsymbol{x}) = \sum_{i=1}^{N}\delta(\boldsymbol{x}), \boldsymbol{x} \in \mathcal{D}^+$ as an empirical distribution, characterized by a probability density function $l(\boldsymbol{x})$, where δ denotes the Dirac delta function (Raju 1982) utilized to convert the discrete offline dataset into a continuous data representation for unified processing. Similarly, the high-risk constraint distributions $h_C(\boldsymbol{x})$, and the high-risk OOD distributions $h_O(\boldsymbol{x})$ are also expressed using a Dirac delta function. Then, we construct the high-risk distribution as $h(\boldsymbol{x}) = h_O(\boldsymbol{x}) + \beta h_C(\boldsymbol{x})$, where $\beta_i = \frac{\exp\sum_j g_{ij}}{\sum_{k=0}^{K}\exp\sum_j g_{kj}}$ can be considered as the degree of constraint violations: while higher β_i steers the model away from infeasible regions, lower values prioritize regression accuracy. g_{ij} means the degree to which the j-th constraint is violated corresponding to the i-th design $\boldsymbol{x}_i$.

The Proposed Method

This section presents the Constrained Offline Optimization via Risk Evaluation and Management (COOREM) method, as illustrated in Figure 2 (a). COOREM consists of three modules. The first module, Adaptive Surrogate Model (ASM), estimates the unknown objective function while reducing the estimations for infeasible designs in $\mathcal{D}^-$. To address the challenges mentioned in COO, we introduce the second module, Composite Risk Evaluation Model (CREM). This module assesses OOD, and constrained risks. ASM and CREM are specifically designed during the training process of COO. We use a three-head DNN with the same input $\boldsymbol{x}$ and three different outputs: the surrogate model head (i.e., ASM), and the dual-energy model heads for OOD and constrained risks (i.e., CREM). Subsequently, the third module, Adaptive Risk Management (ARM), balances these risks throughout the optimization process to obtain stable, high-quality (i.e., high-scoring and feasible) designs. The overall algorithm is detailed in Algorithm 1.

Adaptive Surrogate Model (ASM)

The infeasible designs $\boldsymbol{x}$ in $\mathcal{D}^-$ do not have corresponding scores y in some setting. As a result, the traditional surrogate model, which trains the model $\hat{f}_{\boldsymbol{\theta}}$ by using mean squared error (MSE), cannot be applied. At the same time, overlooking these infeasible designs during training results in terrible performance in practice. Based on this, this paper proposes that ASM incorporate infeasible information into the surrogate model, cf. Figure 2 (b).

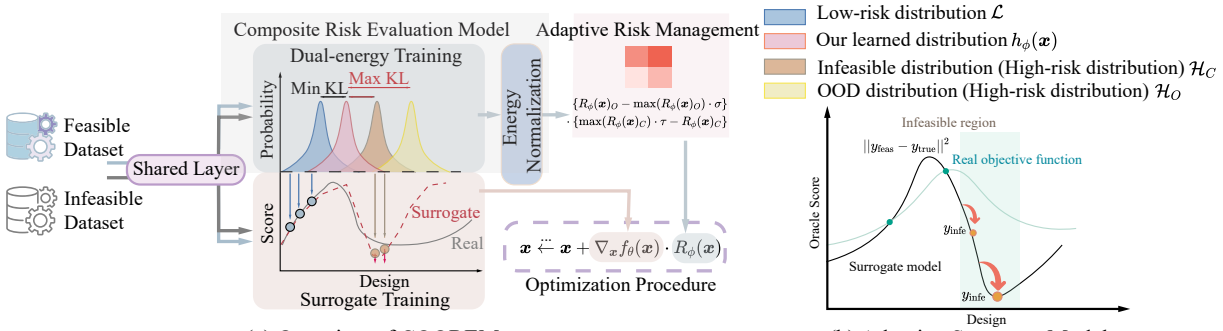
Our core idea is to fit the unknown objective function and reduce the estimations of the infeasible designs $\boldsymbol{x}$ to avoid choosing the infeasible area. Following (Kumar et al. 2022), we incorporate a regularization term into the traditional MSE loss function, which can be formulated as:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}_i, y_i\sim\mathcal{D}^+}[(\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2] - \alpha\mathbb{E}_{\boldsymbol{x}'_i\sim\mathcal{D}^-}[\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x}'_i)]\,. \quad (2)$$

The first term means the standard MSE loss, which is used to fit the surrogate model to the data in $\mathcal{D}^+$. The second term, known as the constraint suppression term, minimizes the expected scores of the surrogate on infeasible designs. Obviously, α plays an indispensable role in determining the behavior of the surrogate. If α is set too low, the optimizer may explore infeasible regions. Conversely, if α is set too high, the estimated values for infeasible designs may become too low, preventing the optimizer from obtaining better designs. Thus, ASM adjusts weights adaptively to differentiate between varying degrees of constraint violations using α.

Composite Risk Evaluation Model (CREM)

Although we have trained the surrogate model from Eq. (2), it is still difficult to achieve high-quality designs in the whole optimization process. That's because designs away from datasets $\mathcal{D}$ inevitably raise the risk of performance degradation (i.e., the optimizer finds OOD or infeasible designs) in the optimization process. To address it, we present CREM



(a) Overview of COOREM (b) Adaptive Surrogate Model

Figure 2: Illustration of adaptive surrogate model (y_{feas} means feasible scores in feasible dataset $\mathcal{D}^+$, y_{infe} means feasible scores in feasible dataset $\mathcal{D}^-$, y_{true} means the true scores in the unknown function) and an overview of COOREM.

to express two different risks, i.e., OOD risk and constrained risk, by training a dual-energy model, and we then control the risks in the next section (Adaptive Risk Management).

Inspired by (Lu et al. 2023), to approximate the low-risk distribution $l(\boldsymbol{x})$ and distance itself from the high-risk distribution $h(\boldsymbol{x})$, an energy-based model (EBM) (Du et al. 2021) is applied to explicitly delineate the risks, which quantifies the likelihood of design $\boldsymbol{x}$'s probability distribution.

$$p_{\boldsymbol{\phi}}(\boldsymbol{x}) = Z(\boldsymbol{\phi})^{-1} \exp(-E_{\boldsymbol{\phi}}(\boldsymbol{x})), \tag{3}$$

where $E_{\boldsymbol{\phi}}(\boldsymbol{x})$ is the energy function parametrized by $\boldsymbol{\phi}$ that maps design $\boldsymbol{x} \in \mathbb{R}^D$ to energy $E_{\boldsymbol{\phi}}(\boldsymbol{x}) \in \mathbb{R}$, i.e., $E_{\boldsymbol{\phi}} : \mathbb{R}^D \to \mathbb{R}$. $Z(\boldsymbol{\phi}) = \int_{\boldsymbol{x}} \exp(-E_{\boldsymbol{\phi}}(\boldsymbol{x}))\, d\boldsymbol{x}$ is the partition function which is a constant w.r.t the variable $\boldsymbol{x}$.

The energy function utilizes Contrastive Divergence (CD) (Hinton 2002) to distinguish between high-risk and low-risk designs. The loss function for training $p_{\boldsymbol{\phi}}(\boldsymbol{x})$ can be expressed as:

$$\mathcal{L}_{\text{CD}_O}(\boldsymbol{\phi}) = \text{KL}\left(l(\boldsymbol{x}) \| p_{\boldsymbol{\phi}_O}(\boldsymbol{x})\right) - \text{KL}\left(h_O(\boldsymbol{x}) \| p_{\boldsymbol{\phi}_O}(\boldsymbol{x})\right), \tag{4}$$

$$\mathcal{L}_{\text{CD}_C}(\boldsymbol{\phi}) = \text{KL}\left(l(\boldsymbol{x}) \| p_{\boldsymbol{\phi}_C}(\boldsymbol{x})\right) - \text{KL}\left(h_C(\boldsymbol{x}) \| p_{\boldsymbol{\phi}_C}(\boldsymbol{x})\right), \tag{5}$$

where $\mathcal{L}_{\text{CD}_O}(\boldsymbol{\phi})$ is the loss of erroneously overestimation and $\mathcal{L}_{\text{CD}_C}(\boldsymbol{\phi})$ is the loss of constraint violations. On the one hand, $\mathcal{L}_{\text{CD}}$ aims to minimize the Kullback-Leibler divergence between low-risk distribution $l(\boldsymbol{x})$ and $p_{\boldsymbol{\phi}}(\boldsymbol{x})$, which drives $p_{\boldsymbol{\phi}}(\boldsymbol{x})$ to approximate the distribution $l(\boldsymbol{x})$. On the other hand, to discourage $p_{\boldsymbol{\phi}}(\boldsymbol{x})$ from adopting the high-risk distribution $h(\boldsymbol{x})$, the divergence between $h(\boldsymbol{x})$—which includes both the high-risk OOD distribution $h_O(\boldsymbol{x})$ and the high-risk constraint distribution $h_C(\boldsymbol{x})$—and $p_{\boldsymbol{\phi}}(\boldsymbol{x})$ is maximized in $\mathcal{L}_{\text{CD}_C}(\boldsymbol{\phi})$ and $\mathcal{L}_{\text{CD}_C}(\boldsymbol{\phi})$.

To satisfy the requirement for the OOD distribution $h_O(\boldsymbol{x})$ in Eq. (4), which is absent in the offline dataset, it's necessary to simulate this distribution. Markov Chain Monte Carlo (MCMC) technique (Welling and Teh 2011) is applied to systematically sample high-risk designs and construct the distribution $h_O(\boldsymbol{x})$. To achieve it, we use Langevin Dynamics $LD_{\boldsymbol{\theta}}$ as the stochastic kernel for MCMC sampling, i.e.,

$$\boldsymbol{x}_k \leftarrow \boldsymbol{x}_{k-1} + \lambda \nabla_{\boldsymbol{x}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_{k-1}) + \boldsymbol{\omega}_k, k = 1, \ldots, K, \tag{6}$$

where $\boldsymbol{w}_k \sim \mathcal{N}(0, \lambda)$, λ is a hyperparameters and K is the maximum Langevin dynamics time step. Besides, $\nabla_{\boldsymbol{x}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_{k-1})$ means the gradient of ASM at $\boldsymbol{x}_{k-1}$.

Algorithm 1: Constrained Offline Optimization via Risk Evaluation and Management (COOREM)

Input: Offline dataset $\mathcal{D}$, learning rate η, maximum Langevin dynamics step K, Langevin dynamics step-size λ, and initial momentum m.

Output: Final solution $\boldsymbol{x}_T$ after optimizing T steps.

Sample OOD high-risk distribution $h_O(\boldsymbol{x})$ and infeasible high-risk distribution $h_C(\boldsymbol{x})$ by Langevin dynamics in Eq. (6) and infeasible offline dataset.
Sample low-risk distribution $l(\boldsymbol{x})$ from feasible offline dataset.
{**Phase 1**: Training Process}
for $i = 1$ in training steps **do**
  ▷ **Adaptive Surrogate Model (ASM)**
  Train Surrogate $\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x})$ on $\mathcal{D}$ using MSE loss with Eq. (2).
  ▷ **Composite Risk Evaluation Model (CREM)**
  Train dual-energy model $E_{\boldsymbol{\phi}_O}(\boldsymbol{x})$ and $E_{\boldsymbol{\phi}_C}(\boldsymbol{x})$ using contrastive divergence loss with Eq. (4) and Eq. (5).
end for
Calculate the risk suppression factor $R_{\boldsymbol{\phi}_O}(\boldsymbol{x})$ and $R_{\boldsymbol{\phi}_C}(\boldsymbol{x})$ with Eq. (8).
{**Phase 2**: Optimization Process}
▷ **Adaptive Risk Management (ARM)**
for $t = 1$ in optimization steps T **do**
  $\boldsymbol{x}_t \leftarrow \boldsymbol{x}_{t-1} + \left(R_{\boldsymbol{\phi}_O}(\boldsymbol{x}) - \sigma R^{max}_{\boldsymbol{\phi}_O}\right)\left(\tau R^{max}_{\boldsymbol{\phi}_C} - R_{\boldsymbol{\phi}_C}(\boldsymbol{x})\right)\nabla_{\boldsymbol{x}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1})$
end for
return $\boldsymbol{x}_T$

The motivation and specific process of Langevin Dynamics $LD_{\boldsymbol{\theta}}$ to simulate high-risk OOD distribution $h_O(\boldsymbol{x})$ are as follows. First of all, as shown in Theorem 1, the reason why Langevin Dynamics $LD_{\boldsymbol{\theta}}$ can be used is that the risk in it gradually increases with each sampling step, and ultimately the return distribution $h_O(\boldsymbol{x}_k)$ is high risk. To achieve it, Langevin Dynamics $LD_{\boldsymbol{\theta}}$, starting from the low-risk distribution $l(\boldsymbol{x})$, does K iterations of gradient ascent to characterize high-risk OOD distribution $h_O(\boldsymbol{x})$.

Theorem 1 (Upper Bound of the Prediction Error in

Langevin Sampling). *In the Langevin dynamics procedure, if the norms* $\|f\|_\infty = \sup\{|f(\boldsymbol{x})| : \boldsymbol{x} \in \mathcal{X}\}$ *and* $\|\hat{f}_{\boldsymbol{\theta}}\|_\infty = \sup\{|\hat{f}(\boldsymbol{x})| : \boldsymbol{x} \in \mathcal{X}\}$ *are defined, the prediction error for* $\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x})$ *over any OOD distribution* $h_O(\boldsymbol{x})$ *is constrained by an upper bound:*

$$\widehat{S}_{\boldsymbol{\theta}}(h_O^k) - S(h_O^k) \leq \widehat{S}_{\boldsymbol{\theta}}(l) - S(l) + C_{f;f_{\boldsymbol{\theta}}} TV(h_O^k; l)\,, \quad (7)$$

where $C_{f;f_{\boldsymbol{\theta}}} = 2(\|\widehat{f}_{\boldsymbol{\theta}}\|_\infty + \|f\|_\infty)$, $\hat{S}_{\boldsymbol{\theta}}(\pi) = \mathbb{E}_{\boldsymbol{x}\sim\pi}[\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x})]$, $S(\pi) = \mathbb{E}_{\boldsymbol{x}\sim\pi}[f(\boldsymbol{x})]$, *and the total variation* $TV(h_O^k; l) = \int_{\mathcal{X}} |h_O(\boldsymbol{x}^k) - l(\boldsymbol{x})| \mathrm{d}\boldsymbol{x}$.

The proof of Theorem 1 can be found in Appendix A. And Appendix A-B can be found in https://github.com/zhuyiyi-123/COOREM. Theorem 1 demonstrates that as $LD_{\boldsymbol{\theta}}$ progresses, the gap between $l(\boldsymbol{x})$ and $h_O(\boldsymbol{x}^k)$ increases, resulting in a larger total variation $TV(h_O^k; l)$. Consequently, the final distribution $h_O(\boldsymbol{x}^k)$ attains the largest upper bound, making it more susceptible to large prediction errors. This is why it is considered the high-risk OOD distribution $h_O(\boldsymbol{x}^k)$.

In Eq.(3), $Z(\boldsymbol{\phi})$ requires integration over the entire input space, making it impractical to compute directly. Thus, it is necessary to identify an alternative approach. Fortunately, favoured by theorem 2, we find that $\mathcal{L}_{\mathrm{CD}}$ can be equivalently represented as $\mathbb{E}_{\boldsymbol{x}\sim l(\boldsymbol{x})}\left[E_{\boldsymbol{\phi}}(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}\sim h(\boldsymbol{x})}\left[E_{\boldsymbol{\phi}}(\boldsymbol{x})\right]$.

Theorem 2 (Equivalent form of $\mathcal{L}_{\mathrm{CD}}(\boldsymbol{\phi})$). *The loss function* $\mathcal{L}_{\mathrm{CD}}(\boldsymbol{\phi})$ *can be equivalently expressed as* $\mathbb{E}_{\boldsymbol{x}\sim l}[E_{\boldsymbol{\phi}}(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim h}[E_{\boldsymbol{\phi}}(\boldsymbol{x})]$, *where* $l(\boldsymbol{x})$ *represents a low-risk distribution and* $h(\boldsymbol{x})$ *represents a high-risk distribution.*

The proof of Theorem 2 is provided in Appendix A. Theorem 2 suggests that training the distribution $h(\boldsymbol{x})$ is equivalent to directly training on the output of $E_{\boldsymbol{\phi}}(\boldsymbol{x})$. This alternative training approach lowers the energy of designs in the low-risk distribution $l(\boldsymbol{x})$ and raises it in the high-risk distribution $h(\boldsymbol{x})$. Consequently, the energy model can estimate the risk of any design from $h(\boldsymbol{x})$ and $l(\boldsymbol{x})$, with the output energy serving as a measure of risk for the designs.

Adaptive Risk Management (ARM)

The core of COO is to find feasible and better designs in the whole optimization. However, despite prior COO methods (Kumar et al. 2022; Huang and Wang 2021) having taken this into account, the issue, i.e., the risk of performance degradation increases, is still available as the optimization process proceeds. To address it, we propose ARM, which includes proposing risk suppression factor and detailing its application in the optimization process.

Risk Suppression Factor of COO. To apply risk in the optimizer and perform stable optimization, we only need to add the risk into the optimization process. In practice, we adopt a normalization method to map energy values $E_{\boldsymbol{\phi}}(\boldsymbol{x})$ in Eq. (3) to a specific range to enhance the model's robustness and ensure consistent performance during optimization. Therefore, the risk suppression factor can be expressed as:

$$R_\phi(\boldsymbol{x}) = \frac{m(E_h(\boldsymbol{x}) - E_\phi(\boldsymbol{x}))}{E_h(\boldsymbol{x}) - E_l(\boldsymbol{x})}\,, \quad (8)$$

where $E_h(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{x}'\sim h(\boldsymbol{x})}\left[E_{\boldsymbol{\phi}}\left(\boldsymbol{x}'\right)\right]$, $E_l(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{x}'\sim l(\boldsymbol{x})}[E_{\boldsymbol{\phi}}\left(\boldsymbol{x}'\right)]$, and m is defined as the initial momentum.

Traditional methods tend to directly apply a gradient ascent with the fixed learning rate for optimization, which may result in finding erroneous overestimation, or infeasible designs; PRIME addresses erroneous overestimation and constrained issues through conservative models, which indiscriminately lower the scores y of OOD and infeasible designs $\boldsymbol{x}$. However, as shown in Figure 1, OOD and infeasible situations should not be treated equally. Equating the two risks can result in an overly conservative model, avoiding the area whenever either risk is high. To address it, ARM incorporates the risk suppression factor $R_\phi(\boldsymbol{x})$ into the gradient ascent optimizer to identify current high-risk states and thus adopt appropriate strategies:

$$\boldsymbol{x}_t \leftarrow \boldsymbol{x}_{t-1} + R_{\boldsymbol{\phi}}(\boldsymbol{x}_{t-1}) \nabla_{\boldsymbol{x}} f_\phi(\boldsymbol{x}_{t-1})\,. \quad (9)$$

$R_\phi(\boldsymbol{x})$ acts similarly to an adaptive step size in online optimization methods like Adam. Unlike online optimizers, which can adjust based on feedback (i.e., active interaction), offline optimizers cannot do it. ARM proposes using risk information from the offline dataset $\mathcal{D}$ for adaptive optimization. The optimizer adjusts based on the current risk.

Specifically, in Figure 1 (a), when neither OOD nor infeasible situations are present (both the OOD risk suppression factor $R_{\boldsymbol{\phi}_O}(\boldsymbol{x})$ and the constrained risk suppression factor $R_{\boldsymbol{\phi}_C}(\boldsymbol{x})$ are high), a standard learning rate is applied. In Figure 1 (b), with low OOD risk and significant constraint violations ($R_{\boldsymbol{\phi}_O}(\boldsymbol{x})$ high, $R_{\boldsymbol{\phi}_C}(\boldsymbol{x})$ low), a larger learning rate is used to quickly move away from infeasible designs. In Figure 1 (c), the optimizer should reverse direction when high OOD risk indicates that low constrained risk may be unreliable, suggesting a retreat. Finally, in Figure 1 (d), the model should avoid high-risk designs by following the gradient in the opposite direction, as halting optimization at this time can degrade design quality. We find that ARCOO has typically reduced the step size to near zero in high-risk regions to prevent obtaining poor-quality designs. However, our approach offers an improvement by not only preventing poor designs but also guiding the optimization process back to low-risk regions. This proactive adjustment ensures that the optimization remains focused on areas with a higher likelihood of success, thereby enhancing overall design quality.

To accomplish this, we introduce a novel risk suppression factor $R_{\boldsymbol{\phi}}(\boldsymbol{x})$ that integrates OOD risk suppression factor $R_{\boldsymbol{\phi}_O}(\boldsymbol{x})$ and constrained risk suppression factor $R_{\boldsymbol{\phi}_C}(\boldsymbol{x})$:

$$R_\phi(\boldsymbol{x}) = \left(R_{\boldsymbol{\phi}_O}(\boldsymbol{x}) - \sigma R_{\boldsymbol{\phi}_O}^{\max}\right)\left(\tau R_{\boldsymbol{\phi}_C}^{\max} - R_{\boldsymbol{\phi}_C}(\boldsymbol{x})\right)\,, \quad (10)$$

where σ and τ are hyperparameters. OOD and constrained risk suppression factors jointly determine both the magnitude and direction of the novel risk suppression factor $R_\phi(\boldsymbol{x})$, which are collectively influenced by the product of the OOD and infeasible risk. $R_{\boldsymbol{\phi}_O}(\boldsymbol{x})^{\max}$ and $R_{\boldsymbol{\phi}_C}(\boldsymbol{x})^{\max}$ are the local maximization of risk in each round. Theorem 3 has proved that $R_\phi(\boldsymbol{x})$ has an upper bound that can control risk within an appropriate range.

Theorem 3 (Upper Bound of the Risk Suppression Factor). *Given a design* $\boldsymbol{x}$, *consider a Gaussian distribution* $\mathcal{N}(\boldsymbol{x}; \sigma)$ *where* σ *is the standard deviation and* $c = m^2 \cdot (\tau - 1) \cdot (1 - \sigma)$, *if* $\|E\|_\infty$ *exists, then* $R_{\boldsymbol{\phi}}(\boldsymbol{x})$ *is upper bounded:*

$$R_{\boldsymbol{\phi}}(\boldsymbol{x}) \leq c \cdot \frac{TV(h_O^k(\boldsymbol{x}); \mathcal{N}(\boldsymbol{x}; \sigma))}{TV(h_O^k(\boldsymbol{x}); P_l(\boldsymbol{x}))} \cdot \frac{TV(h_C^k(\boldsymbol{x}); \mathcal{N}(\boldsymbol{x}; \sigma))}{TV(h_C^k(\boldsymbol{x}); P_l(\boldsymbol{x}))}\,. \quad (11)$$

The proof of Theorem 3 can be found in Appendix A. Theorem 3 reveals the distance constrains the risk suppression factor from the high-risk distribution $h(\boldsymbol{x})$, σ, and τ on each optimization process. When the current designs approach the high-risk distribution $h(\boldsymbol{x})$, the bound contracts and mitigates the risk, resulting in a decrease. The new risk suppression factor $R_\phi(\boldsymbol{x})$ adjusts adaptively based on risk. Specifically, the higher risk suppression factor $R_\phi(\boldsymbol{x})$, the lower the risk. A low $R_{\boldsymbol{\phi}_O}(\boldsymbol{x})$ indicates a high OOD risk. Given this high risk, following the gradient is advisable, irrespective of other factors. Consequently, σ is set to a minimal value, such as 0.1, which prompts a retreat when the risk falls below this threshold, i.e., $R_{\boldsymbol{\phi}_O}(\boldsymbol{x}) < \sigma R_{\boldsymbol{\phi}_O}(\boldsymbol{x})^{\max}$. When a constraint is violated, and the OOD risk is low, as indicated by $\tau R_{\boldsymbol{\phi}_C}(\boldsymbol{x})^{\max} < R_{\boldsymbol{\phi}_C}(\boldsymbol{x})$, increasing the step size is beneficial to move away from infeasible designs swiftly. Conversely, if the risk is high, a larger step size helps revert to feasible designs. Furthermore, a moderate step size is preferable when both risk suppression factors (i.e., $R_{\boldsymbol{\phi}_O}(\boldsymbol{x})$ and $R_{\boldsymbol{\phi}_C}(\boldsymbol{x})$) are high. Consequently, we set τ greater than 1 (e.g., 1.1) to adjust the action of the gradient.

Experiments

This section first introduces the experimental setup, including baselines and tasks. Extensive experiments are conducted on these tasks to answer the following questions. The code of this paper is available at https://github.com/zhuyiyi-123/COOREM.

Q1: How does COOREM perform when compared with online and offline state-of-the-art (SOTA) algorithms in terms of optimality and stability?

Q2: Can COOREM mitigate OOD and constraint issues?

Q3: How do ASM and ARM contribute to the performance of COOREM respectively?

Q4: How do the hyperparameters, i.e., the step of Langevin Dynamic K, risk control σ and τ, affect COOREM?

Task Description

We conduct experiments on two gtopx space mission tasks: Cassini 1 and Cassini1-MINLP (Schlueter et al. 2021) and three CEC tasks (Kumar et al. 2020): Three-bar truss design problem, Process synthesis problem, Welded beam design.

Table 1 below shows the number of decision variables, the number of constraints, and the feasible optimal objective function values of the five tasks. A detailed task description and implementation details can be found in Appendix B.

Tasks	D	g	$f(\boldsymbol{x}^*)$
Cassini 1	6	4	4.9307
Cassini 1-MINLP	10	4	3.5007
Three-bar	2	3	263.8958
Process synthesis	2	2	2.0000
Welded beam design	4	5	1.6702

Table 1: Details of experiments' dataset, D and g means the number of decision variables and constraints, and $f(\boldsymbol{x}^*)$ is the best feasible objective function value.

Experimental Setup

This subsection introduces the experimental setup including compared methods, optimality indicator, and stability indicator. The details are provided in Appendix B.

Baselines and Comparisons. We compare COOREM against four COO methods and three constrained online optimization methods. All the experiments are independently repeated with 10 seeds. In all the experiments, we provide the same dataset to all COO methods, while the online methods can actively select which designs to query.

Constrained offline optimization methods:

- DE-PF(S) (Huang and Wang 2021) is a COO method, which uses static penalty functions to handle constraints.
- DE-PF(A) (Huang and Wang 2021) is a COO method, which uses adaptive penalty functions to handle.
- DE-SFP(A) (Huang and Wang 2021) is a COO method, which uses superiority of feasible designs to handle.
- PRIME (Kumar et al. 2022) utilizes a conservative model to solve OOD and constrained issues.

Constrained online optimization methods:

- SCBO (Eriksson and Poloczek 2021) is a scalable constrained Bayesian optimization.
- CEO is a constrained evolutionary optimization, which is implemented by Scikit-opt.
- LLM (Ito and Kunisch 2008) is the traditional Lagrange Multiplier Method.

Optimality Indicator (OI). Evaluating the performance of COOREM can be a challenging task as the result of stopping at a certain step does not indicate completely the optimal performance of the algorithm. Comparing algorithm A (current constrained offline optimization algorithm) with $\mathrm{A_{con}}$ (the most conservative baseline, i.e., optimal design in the offline feasible dataset), OI serves as a measure, quantifying the degree to which A exceeds $\mathrm{A_{con}}$'s performance. It can be formally defined as: $\mathrm{OI(A)} = \frac{S_\mathrm{A}}{S_{\mathrm{A_{con}}}}$, where S_A denotes the shared area under the curve of algorithm A, and $S_{\mathrm{A_{con}}}$ signifies the area under the curve of algorithm $\mathrm{A_{con}}$.

Stability Indicator (SI). Evaluating the stability of COOREM is an important task as we don't know at which step the algorithm stops. By comparing the optimal design obtained by algorithm A at each step of optimization with those of an ideal algorithm $\mathrm{A_{ideal}}$ (i.e., the optimal design found in the whole optimization), we measure the stability of algorithm A. The higher SI, the closer the designs generated by algorithm A are to the optimal design found in the optimization processes. The formula can be expressed as $\mathrm{SI(A)} = \frac{S_\mathrm{A}}{S_{\mathrm{A_{ideal}}}}$, where $S_{\mathrm{A_{ideal}}}$ is the area under the hypothetical ideal algorithm $\mathrm{A_{ideal}}$.

Experimental Results

We focus on OI and SI for COO methods due to their inability to interact with unknown functions, preventing real-time quality assessment. This makes maintaining stability and

	Tasks	Three-bar			Process synthesis			Cassini 1			C-MINLP			Welded beam		
Online	SCBO	-264.50			-2.00			-282.00*			-147.60			-1.87		
	CEO	-263.90			-2.00			-7.27			-9.25			-2.10		
	Lagrange	-280.91			-2.89			-			-			-7.82		
		Best	OI	SI	Best	OI	SI	Best	OI	SI	Best	OI	SI	Best	OI	SI
	$\boldsymbol{x}_{\text{OFF}}$	-265.89			3.20			-11.92			-22.44			-7.08		
Offline	DE-PF(S)	-273.65	1.16	1.16	-2.82	0.89	1.08	-	835.41	835.41	-	443.77	443.77	-8.83	1.67	1.67
	DE-PF(A)	-272.41	1.08	1.08	-2.29	0.95	1.03	-	835.41	835.41	-	443.77	443.77	-8.72	2.25	2.25
	DE-SPF(A)	-280.99	1.07	1.07	-2.50	0.94	1.01	-	835.41	835.41	-	443.77	443.77	-9.4	1.21	1.27
	PRIME	-293.59*	37.15	37.15	-2.22	3006.86	4381.60	-	835.41	835.41	-	443.77	443.77	-	1406.50	1406.50
	COOREM	**-264.72**	**0.89**	**1.01**	**-2.09**	**0.65**	**1.03**	**-12.92**	**1.01**	**1.01**	**-7.84**	**0.79**	**1.08**	**-2.93**	**0.82**	**1.05**

Table 2: Comparisons on the quality of final designs with constrained online optimization and COO methods. $\boldsymbol{x}_{\text{OFF}}$ means the best feasible design in offline dataset. The value marked in * indicates that the feasibility rate is lower than 80% during 10 independent repetitions. "-" in the text means feasible designs cannot be found. In this case, we default to rating the quality of the design as extremely poor and assigning a value of 10,000.

high-quality predictions essential for good results. In contrast, online optimization allows for real-time solution quality evaluation and model adjustment to improve outcomes. Since the standard deviation of online and offline algorithms is relatively small, the experimental results in this paper only provide the mean value.

Optimality and Stability Performance (Q1). Table 2 shows that COOREM almost outperforms all COO methods and some online methods. In the CEC tasks, which are relatively easy to optimize due to low search dimensions and few constraints, nearly all online and offline methods, except PRIME, can find feasible and stable high-scoring designs in Table 2. However, in more challenging gtopx space mission tasks, online optimization algorithms SCBO and CEO can find feasible designs, but not only are the designs found by SCBO not good, but also prior COO methods fail to find feasible designs even in the first few steps of optimization, and this results in large values of OI and SI. For example, previous COO methods could not find a feasible design on the Cassini 1 task, so they assign a poor value, resulting in their OI and SI being equal and having a very high value of 835.41. COOREM successfully finds high-quality designs through effective risk control strategies. In addition, our research finds that compared with traditional conservative constrained online optimization methods, methods using evolutionary strategies perform better.

Ablation Study (Q2 & Q3). To understand how COOREM addresses the OOD and constrained problems jointly, and how the ASM and ARM components affect the performance of COOREM, we conduct an ablation study on the Welded task. Specifically, we compare COOREM with the following variants:

- COOREM w/o CSM - A surrogate model with no control over constraints.
- COOREM w/o ASM - A proxy model with static control over constraints.
- COOREM w/o ARM - A normal gradient ascent.
- COOREM w/o FARM - A gradient ascent with learning rate $R_{\phi_O}(\boldsymbol{x})R_{\phi_C}(\boldsymbol{x})$.
- COOREM w/o ARMO - A gradient ascent algorithm that only cares about OOD risk.
- COOREM w/o ARMC - A gradient ascent algorithm that only cares about constrained risk.

Such comparisons help reveal the impact of each component on overall performance.

- Compared to COOREM, COOREM w/o CSM and ASM can only get a small improvement because although we use risk to know the operation of the gradient, it tends to go to those high-scoring but infeasible designs, even if we handle the constraints like COOREM w/o ASM.
- Compared to COOREM, COOREM w/o ARM does not improve performance due to the overestimation of high-risk OOD designs later in the optimization process. The algorithm often gets trapped in infeasible areas, failing to distinguish these from feasible ones.
- Compared to COOREM, COOREM w/o ARMO and ARMC will stop at the beginning of optimization as they can only handle one bad situation and are easily trapped in another bad situation.

Additional ablation analysis is in Appendix B.

Hyperparameter Analysis (Q4). We also conduct hyperparameter analysis on the impact of the step of Langevin Dynamics K, σ and τ in the risk control. COOREM exhibits robustness and low sensitivity to hyperparameters. The detailed results are available in Appendix B due to space limitation.

Conclusion

This paper presents a constrained offline black-box optimization method called COOREM, which uses the energy-based model to assess risks during the optimization process and introduces a novel strategy to control them. We also propose the adaptive surrogate model, which can avoid sampling infeasible areas. Compared with existing COO methods, COOREM provides superior performance. Notably, dynamically adjusting the step size and keeping it stable during the optimization process can help the surrogate focus more on improving performance rather than being deliberately conservative. COOREM is an attempt to take a solid step forward in finding stable, good, and feasible designs. In the future, developing strategies for surrogate models with better generalization performance as a prior is expected.

Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive and valuable comments. The authors also would like to thank Kelei Wu for the reliable support. The algorithms and datasets in this paper do not involve any ethical issue. This work is supported by the National Natural Science Foundation of China (No. 62476091, No. 62106076).

References

Brookes, D. H.; Park, H.; and Listgarten, J. 2019. Conditioning by Adaptive Sampling for Robust Design. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, 773–782. Long Beach,CA.

Chen, C.; Zhang, Y.; Fu, J.; Liu, X. S.; and Coates, M. 2022. Bidirectional Learning for Offline Infinite-Width Model-Based Optimization. In *Advances in Neural Information Processing Systems 35*, volume 35, 29454–29467. New Orleans, LA.

Chen, C.; Zhang, Y.; Liu, X.; and Coates, M. 2023. Bidirectional Learning for Offline Model-based Biological Sequence Design. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, 5351–5366. Honolulu, HI.

Du, Y.; Li, S.; Tenenbaum, J. B.; and Mordatch, I. 2021. Improved Contrastive Divergence Training of Energy-Based Models. In *Proceedings of the 38th International Conference on Machine Learning*, 2837–2848. Virtual.

Eriksson, D.; and Poloczek, M. 2021. Scalable Constrained Bayesian Optimization. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, volume 130, 730–738. Virtual.

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, 315–323.

Gong, Y.; Zhong, Y.; and Huang, H. 2024. Offline Data-Driven Optimization at Scale: A Cooperative Coevolutionary Approach. *IEEE Transactions on Evolutionary Computation*, 28(6): 1809–1823.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27*, volume 27, 2672–2680. Quebec, Canada.

Gu, Y.; Wu, Y.; Lu, H.; Lu, X.; Qian, H.; Zhou, J.; and Zhou, A. 2024. LASCA: A Large-Scale Stable Customer Segmentation Approach to Credit Risk Assessment. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5006–5017. Barcelona, Spain.

Hinton, G. E. 2002. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8): 1771–1800.

Huang, P.; and Wang, H. 2021. Comparative Empirical Study on Constraint Handling in Offline Data-Driven Evolutionary Optimization. *Applied Soft Computing*, 110: 107603.

Ito, K.; and Kunisch, K. 2008. *Lagrange Multiplier Approach to Variational Problems and Applications*. SIAM.

Jin, Y.; Wang, H.; and Sun, C. 2021. *Data-driven evolutionary optimization*. Springer.

Kingma, D. P.; and Welling, M. 2014. Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations*. Banff, Canada.

Kumar, A.; and Levine, S. 2020. Model Inversion Networks for Model-Based Optimization. In *Advances in Neural Information Processing Systems 33*, volume 33, 5126–5137. Virtual.

Kumar, A.; Wu, G.; Ali, M. Z.; Mallipeddi, R.; Suganthan, P. N.; and Das, S. 2020. A Test-Suite of Non-Convex Constrained Optimization Problems from the Real-World and Some Baseline Results. *Swarm and Evolutionary Computation*, 56: 100693.

Kumar, A.; Yazdanbakhsh, A.; Hashemi, M.; Swersky, K.; and Levine, S. 2022. Data-Driven Offline Optimization for Architecting Hardware Accelerators. In *Proceedings of the 10th International Conference on Learning Representations*. Virtual.

Liao, T.; Wang, G.; Yang, B. H.; Lee, R.; Pister, K. S. J.; Levine, S.; and Calandra, R. 2019. Data-efficient Learning of Morphology and Controller for a Microrobot. In *2019 International Conference on Robotics and Automation*, 2488–2494. Montreal, Canada.

Lu, H.; Qian, H.; Wu, Y.; Liu, Z.; Zhang, Y.-L.; Zhou, A.; and Yu, Y. 2023. Degradation-Resistant Offline Optimization via Accumulative Risk Control. In *Proceedings of the 26th European Conference on Artificial Intelligence*. Krakow, Poland.

Qi, H.; Su, Y.; Kumar, A.; and Levine, S. 2022. Data-Driven Offline Decision-Making via Invariant Representation Learning. In *Advances in Neural Information Processing Systems 35*, volume 35, 13226–13237. New Orleans, LA.

Raju, C. 1982. Products and compositions with the Dirac delta function. *Journal of Physics A: Mathematical and General*, 15(2): 381.

Schlueter, M.; Neshat, M.; Wahib, M.; Munetomo, M.; and Wagner, M. 2021. GTOPX Space Mission Benchmarks. *SoftwareX*, 14: 100666.

Trabucco, B.; Geng, X.; Kumar, A.; and Levine, S. 2022. Design-Bench: Benchmarks for Data-Driven Offline Model-Based Optimization. In *Proceeding of the 39th International Conference on Machine Learning*, 21658–21676. Baltimore, Maryland.

Trabucco, B.; Kumar, A.; Geng, X.; and Levine, S. 2021. Conservative Objective Models for Effective Offline Model-Based Optimization. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, 10358–10368. Virtual.

Welling, M.; and Teh, Y. W. 2011. Bayesian Learning via Stochastic Gradient Langevin Dynamics. In *Proceedings of the 28th International Conference on Machine Learning*, 681–688. Bellevue, Washington: Omnipress.

Xue, K.; Tan, R.; Huang, X.; and Qian, C. 2024. Offline Multi-Objective Optimization. In *Proceedings of the 41st International Conference on Machine Learning*. Vienna, Austria.

Yu, S.; Ahn, S.; Song, L.; and Shin, J. 2021. RoMA: Robust Model Adaptation for Offline Model-based Optimization. In *Advances in Neural Information Processing Systems 34*, volume 34, 4619–4631. Virtual.