Bidirectional Learning for Offline Model-based Biological Sequence Design

Can (Sam) Chen [1 2] Yingxue Zhang [3] Xue Liu [1] Mark Coates [1]

Abstract

Offline model-based optimization aims to maximize a black-box objective function with a static dataset of designs and their scores. In this paper, we focus on biological sequence design to maximize some sequence score. A recent approach employs bidirectional learning, combining a forward mapping for exploitation and a backward mapping for constraint, and it relies on the neural tangent kernel (NTK) of an infinitely wide network to build a proxy model. Though effective, the NTK cannot learn features because of its parametrization, and its use prevents the incorporation of powerful pre-trained Language Models (LMs) that can capture the rich biophysical information in millions of biological sequences. We adopt an alternative proxy model, adding a linear head to a pre-trained LM, and propose a linearization scheme. This yields a closed-form loss and also takes into account the biophysical information in the pre-trained LM. In addition, the forward mapping and the backward mapping play different roles and thus deserve different weights during sequence optimization. To achieve this, we train an auxiliary model and leverage its weak supervision signal via a bi-level optimization framework to effectively learn how to balance the two mappings. Further, by extending the framework, we develop the first learning rate adaptation module *Adaptive-η*, which is compatible with all gradient-based algorithms for offline model-based optimization. Experimental results on DNA/protein sequence design tasks verify the effectiveness of our algorithm. Our code is available here.

[1]McGill University [2]Mila - Quebec AI Institute [3]Huawei Noah's Ark Lab. Correspondence to: Can (Sam) Chen <can.chen@mila.quebec>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

Offline model-based optimization aims to maximize a black-box objective function with a static dataset of designs and their scores. This offline setting is realistic since in many real-world scenarios we do not have interactive access to the ground-truth evaluation. The design tasks of interest include material, aircraft, and biological sequence (Trabucco et al., 2021). In this paper, we focus on biological sequence design, including DNA/protein sequence, with the goal of maximizing some specified property of these sequences.

A wide variety of methods have been proposed for biological sequence design, including evolutionary algorithms (Sinai et al., 2020; Ren et al., 2022), reinforcement learning methods (Angermueller et al., 2019), Bayesian optimization (Terayama et al., 2021), search/sampling using generative models (Brookes et al., 2019; Chan et al., 2021), and GFlowNets (Jain et al., 2022). Recently, gradient-based techniques have emerged as an effective alternative (Trabucco et al., 2021). These approaches first train a deep neural network (DNN) on the static dataset as a proxy and then obtain the new designs by directly performing gradient ascent steps on the existing designs. Such methods have been widely used in biological sequence design (Norn et al., 2021; Tischer et al., 2020; Linder & Seelig, 2020). One obstacle is the out-of-distribution issue, where the trained proxy model is inaccurate for the newly generated sequences.

To mitigate the out-of-distribution issue, recent work proposes regularization of the model (Trabucco et al., 2021; Yu et al., 2021; Fu & Levine, 2021) or the design itself (Chen et al., 2022c). The first category focuses on training a better proxy by introducing inductive biases such as robustness (Yu et al., 2021). The second category introduces bidirectional learning (Chen et al., 2022c), which consists of a forward mapping and a backward mapping, to optimize the design directly. Specifically, the backward mapping leverages the high-scoring design to predict the static dataset and vice versa for the forward mapping, which distills the information of the static dataset into the high-scoring design. This approach achieves state-of-the-art performances on a variety of tasks. Though effective, the proposed bidirectional learning relies on the neural tangent kernel (NTK) of an infinite-width model to yield a closed-form loss, which is a key component of its successful operation. The NTK cannot learn features due to its parameterization (Yang & Hu, 2021) and thus the bidirectional learning cannot incorporate the wealth of biophysical information from Language Models (LMs) pre-trained over a vast corpus of unlabelled sequences (Elnaggar et al., 2021; Ji et al., 2021).

To solve this issue, we construct a proxy model by combining a finite-width pre-trained LM with an additional layer. We then linearize the resultant proxy model, inspired by the recent progress in deep linearization (Achille et al., 2021; Dukler et al., 2022). This scheme not only yields a closed-form loss but also exploits the rich biophysical information that has been distilled in the pre-trained LM. In addition, the forward mapping encourages exploitation in the sequence space and the backward mapping serves as a constraint to mitigate the out-of-distribution issue. It is vital to maintain an appropriate balance between exploitation and constraint, and this can vary across design tasks as well as during the optimization process. We introduce a hyperparameter γ to control the balance. However, how to properly select γ is challenging because of the problem's offline optimization nature. Thus, we develop a bi-level optimization framework *Adaptive*-γ. In this framework, we train an auxiliary model and leverage its weak supervision signal to effectively update γ. To sum up, we propose ***BI**directional learning for model-based **B**iological sequence design* (**BIB**).

Since the offline nature prohibits standard cross-validation strategies for hyperparameter tuning, all current gradient-based offline model-based algorithms preset the learning rate η. There is a danger of poor selection, and to address this, we propose an *Adaptive*-η module, which effectively adapts the learning rate η via the weak supervision signal from the trained auxiliary model. To the best of our knowledge, *Adaptive*-η is the first learning rate adaptation module for gradient-based algorithms for offline model-based optimization. We discuss the relationship between the *Adaptive* modules and previous hyperparameter optimization work in Sec. 5. Experiments on DNA and protein sequence design tasks verify the effectiveness of BIB and *Adaptive*-η.

To summarize, we propose a novel, highly effective and robust biological sequence design method which consists of deep linearization to incorporate rich biophysical information from the pre-trained model and *Adaptive-* modules to reduce the sensitivity of the hyper-parameter choice. Our contributions are three-fold:

- Instead of adopting the NTK, we construct a proxy model by combining a pre-trained biological LM with an additional trainable layer. We then linearize the proxy model, leveraging the recent progress on deep linearization. This yields a closed-form loss computation in bidirectional learning and allows us to exploit the rich biophysical information distilled into the LM via pre-training over millions of biological sequences.
- We propose a bi-level optimization framework *Adaptive*-γ where we leverage weak signals from an auxiliary model to achieve a satisfactory trade-off between design exploitation and constraint.
- We further extend this bi-level optimization framework to *Adaptive*-η. As the first learning rate tuning scheme in offline model-based optimization, *Adaptive*-η allows learning rate adaptation for any gradient-based algorithm.

2. Preliminaries

2.1. Offline Model-based Optimization

Offline model-based optimization aims to find a design $\boldsymbol{X}$ to maximize some unknown objective $f(\boldsymbol{X})$. This can be formally written as,

$$\boldsymbol{X}^* = \arg\max_{\boldsymbol{X}} f(\boldsymbol{X})\,, \tag{1}$$

where we have access to a size-N dataset $\mathcal{D} = \{(\boldsymbol{X}_1, y_1)\}, \cdots, \{(\boldsymbol{X}_N, y_N)\}$ with $\boldsymbol{X}_i$ representing a certain design and y_i denoting the design score. In this paper, $\boldsymbol{X}_i$ represents a biological sequence design, including DNA and protein sequences, and y_i represents a property of the biological sequence such as the fluorescence level of the green fluorescent protein (Sarkisyan et al., 2016).

2.2. Biological Sequence Representation

Following (Norn et al., 2021; Killoran et al., 2017; Linder & Seelig, 2021), we adopt the position-specific scoring matrix to represent a length-L protein sequence as $\boldsymbol{X} \in \mathbb{R}^{L\times 20}$, where 20 represents 20 different kinds of amino acids. For a real-world protein, $\boldsymbol{X}[\boldsymbol{l},:]$ $(0 \leq l \leq L-1)$ is a one-hot vector denoting one kind of amino acid. During optimization, $\boldsymbol{X}[\boldsymbol{l},:]$ is a continuous vector and $softmax(\boldsymbol{X}[\boldsymbol{l},:])$ represents the probability distribution of all 20 amino acids in the position l. Similarly, for a DNA sequence, we have $\boldsymbol{X} \in \mathbb{R}^{L\times 4}$ where 4 represents 4 different DNA bases.

The protein sequence $\boldsymbol{X}$ is fed into the embedding layer of the LM, which produces the embedding,

$$\boldsymbol{e} = EMB(softmax(\boldsymbol{X}))\,. \tag{2}$$

The main block of the LM takes $\boldsymbol{e}$ as input and outputs biophysical features. The DNA LM, which adopts the k-mer representation, is a little different from protein LMs. See Appendix 7.1 for details.

2.3. Gradient Ascent on Sequence

A common approach to the posed offline model-based optimization is to train a proxy $f_{\boldsymbol{\theta}}(\boldsymbol{X})$ on the offline dataset,

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}(f_{\boldsymbol{\theta}}(\boldsymbol{X}_i) - y_i)^2\,. \tag{3}$$

Then we can obtain the high-scoring design $\boldsymbol{X}_h$ by T gradient ascent steps:

$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t + \eta \nabla_{\boldsymbol{X}} f_{\boldsymbol{\theta}^*}(\boldsymbol{X})|_{\boldsymbol{X}=\boldsymbol{X}_t}\,, \quad \text{for } t \in [0, T-1]\,, \tag{4}$$

where the high-scoring design $\boldsymbol{X}_h$ can be obtained as $\boldsymbol{X}_T$.

Considering the discrete nature of biological sequences, the input of $f_{\boldsymbol{\theta}}(\cdot)$ should be discrete one-hot vectors. Following (Norn et al., 2021), we can perform the following conversion and predict the score via:

$$\hat{\boldsymbol{X}}_i = softmax(\boldsymbol{X}_i)\,, \tag{5}$$

$$\boldsymbol{Z}_i = onehot(argmax(\hat{\boldsymbol{X}}_i))\,, \tag{6}$$

$$\hat{y} = f_{\boldsymbol{\theta}}(\boldsymbol{Z}_i)\,. \tag{7}$$

Then the gradient regarding $\boldsymbol{X}_i$ can be approximated as,

$$\frac{df_{\boldsymbol{\theta}}(\boldsymbol{Z}_i)}{d\boldsymbol{x}_i} \approx \frac{df_{\boldsymbol{\theta}}(\boldsymbol{Z}_i)}{d\boldsymbol{z}_i}\frac{d\hat{\boldsymbol{x}}_i}{d\boldsymbol{x}_i}\,, \tag{8}$$

where we unroll the matrices $\boldsymbol{X}_i$, $\hat{\boldsymbol{X}}_i$ and $\boldsymbol{Z}_i$ as vectors $\boldsymbol{x}_i$, $\hat{\boldsymbol{x}}_i$ and $\boldsymbol{z}_i$ for notational convenience. This approximation allows us to use backpropagation directly from the proxy to the sequence design $\boldsymbol{X}_i$. For brevity, we will still use $f_{\boldsymbol{\theta}}(\boldsymbol{X_i})$ to represent the proxy.

2.4. Bidirectional Learning

As shown in Figure 1, bidirectional learning (Chen et al., 2022c), consists of two mappings: the forward mapping leverages the static dataset ($\boldsymbol{X_l}$, $\boldsymbol{y_l}$) to predict the score y_h of the high-scoring design $\boldsymbol{X}_h$, and the backward mapping leverages the high-scoring design data ($\boldsymbol{X}_h$, y_h) to predict the static dataset ($\boldsymbol{X_l}$, $\boldsymbol{y_l}$). The forward mapping loss is

$$\mathcal{L}_{l2h}(\boldsymbol{X}_h) = \|y_h - f^l_{\boldsymbol{\theta}^*}(\boldsymbol{X}_h)\|^2\,, \tag{9}$$

where $\boldsymbol{\theta}^*$ is given by

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \|\boldsymbol{y}_l - f^l_{\boldsymbol{\theta}}(\boldsymbol{X}_l)\|^2 + \beta\|\boldsymbol{\theta}\|^2\,, \tag{10}$$

where $\beta > 0$ is a regularization parameter. The backward mapping loss can be written as

$$\mathcal{L}_{h2l}(\boldsymbol{X}_h) = \|\boldsymbol{y}_l - f^h_{\boldsymbol{\theta}^*(\boldsymbol{X}_h)}(\boldsymbol{X}_l)\|^2\,, \tag{11}$$

where $\boldsymbol{\theta}^*(\boldsymbol{X}_h)$ is given by

$$\boldsymbol{\theta}^*(\boldsymbol{X}_h) = \arg\min_{\boldsymbol{\theta}} \|y_h - f^h_{\boldsymbol{\theta}}(\boldsymbol{X}_h)\|^2 + \beta\|\boldsymbol{\theta}\|^2\,. \tag{12}$$

The high-scoring design $\boldsymbol{X}_h$ is optimized against the bidirectional learning loss $\mathcal{L}(\boldsymbol{X}_h) = \mathcal{L}_{l2h}(\boldsymbol{X}_h) + \mathcal{L}_{h2l}(\boldsymbol{X}_h)$.

2.5. Deep Linearization

The deep linearization (Mu et al., 2020) is a first-order Taylor expansion of the proxy model with respect to its parameters. As shown in (Bai et al., 2020), for any neural network $f_{\boldsymbol{\theta}}(\boldsymbol{X})$ and a given initialization $\boldsymbol{\theta}_0$, assuming sufficient smoothness, we can expand $f_{\boldsymbol{\theta}}(\boldsymbol{X})$ around $\boldsymbol{\theta}_0$ as

$$f_{\boldsymbol{\theta}}(\boldsymbol{X}) = f_{\boldsymbol{\theta}_0}(\boldsymbol{X}) + \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_0}(\boldsymbol{X})(\boldsymbol{\theta}-\boldsymbol{\theta}_0) + o(\|\boldsymbol{\theta}-\boldsymbol{\theta}_0\|). \tag{13}$$

Since the fine-tuning does not significantly change $\boldsymbol{\theta}_0$, the term $o(\|\boldsymbol{\theta}-\boldsymbol{\theta}_0\|)$ can be neglected and thus we have the local approximation as the deep linearization:

$$f_{\boldsymbol{\theta}}(\boldsymbol{X}) \approx f_{\boldsymbol{\theta}_0}(\boldsymbol{X}) + \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_0}(\boldsymbol{X})(\boldsymbol{\theta}-\boldsymbol{\theta}_0). \tag{14}$$

3. Method

In this section, we first illustrate how to leverage deep linearization to compute the bidirectional learning loss in a closed form. Subsequently, we introduce a hyperparameter γ to control the balance between the forward mapping and the backward mapping. We then develop a novel bi-level optimization framework *Adaptive*-γ, which leverages a weak supervision signal from an auxiliary model to effectively update γ. Last but not least, we extend this framework to *Adaptive*-η, which enables us to adapt the learning rate η for all gradient-based offline model-based algorithms. In summary, we introduce a novel, highly effective, and robust method for biological sequence design, using deep linearization to incorporate rich biophysical information from the pre-trained model and *Adaptive*- modules to reduce hyperparameter sensitivity. We summarize it in Algorithm 1.

3.1. Deep Linearization for Bidirectional Learning

In bidirectional learning, the backward mapping loss is intractable for a finite neural network, so Chen et al. (2022c) employ a neural network with infinite width, which yields a closed-form loss via the NTK. This however makes it impossible to incorporate the rich biophysical information that has been distilled into a pre-trained LM (Yang & Hu, 2021). Considering this, we construct a proxy model by combining a finite-width pre-trained LM with an additional layer. We then linearize the resultant proxy model, inspired by the recent progress in deep linearization which has established that an overparameterized DNN model is close to its linearization (Achille et al., 2021; Dukler et al., 2022).

Denote by $\boldsymbol{\theta_0} = (\boldsymbol{\theta_{pt}}, \boldsymbol{\theta^{lin}_{init}}) \in \mathcal{R}^{D\times 1}$ the proxy model parameters derived by combining the parameters of the pre-trained LM $\boldsymbol{\theta_{pt}}$ and a random initialization of the linear layer $\boldsymbol{\theta^{lin}_{init}}$. In this paper, we adopt the pre-trained DNABERT (Ji et al., 2021) and Prot-BERT (Elnaggar et al., 2021) models, and compute the average of token embeddings as the extracted feature, which is fed into the linear layer to build the proxy. We also study how our method

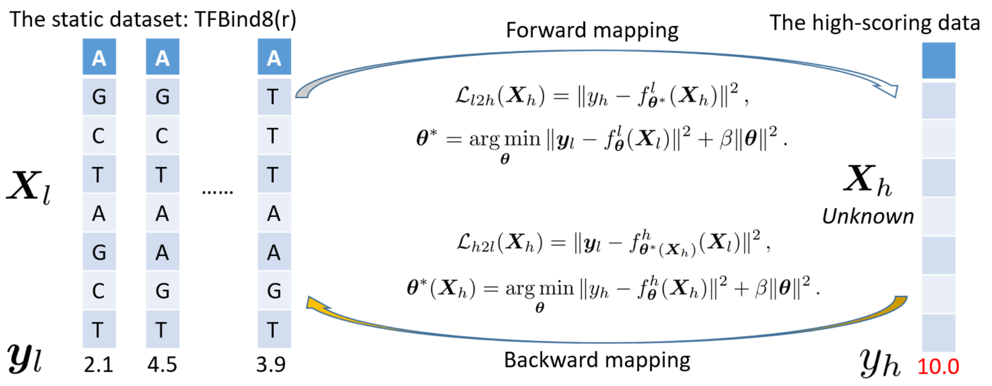

The static dataset: TFBind8(r)
Forward mapping
The high-scoring data
X_l
$\mathcal{L}_{l2h}(X_h) = \|y_h - f^l_{\theta^*}(X_h)\|^2$,
$\theta^* = \arg\min_\theta \|y_l - f^l_\theta(X_l)\|^2 + \beta\|\theta\|^2$.
X_h
Unknown
$\mathcal{L}_{h2l}(X_h) = \|y_l - f^h_{\theta^*(X_h)}(X_l)\|^2$,
$\theta^*(X_h) = \arg\min_\theta \|y_h - f^h_\theta(X_h)\|^2 + \beta\|\theta\|^2$.
Backward mapping
y_l
2.1
4.5
3.9
y_h
10.0

Figure 1. Illustration of bidirectional learning Chen et al. (2022c) where (X_l, y_l) denotes the static dataset, y_h is a large predefined target score and X_h is the high-scoring design we aim to find.

performs as a function of the pre-trained LM quality in Appendix 7.2. Then we can construct a linear approximation for the proxy model:

$$f_{\theta}(X) \approx f_{\theta_0}(X) + \triangledown_{\theta} f_{\theta_0}(X) \cdot (\theta - \theta_0)\,, \tag{15}$$

where $f_{\theta}(X), f_{\theta_0}(X) \in \mathcal{R}$, $\triangledown_{\theta} f_{\theta_0}(X) \in \mathcal{R}^{1\times D}$ and $\triangledown_{\theta} f_{\theta_0}(X) \in \mathcal{R}^{D\times 1}$. Intuitively, if the fine-tuning does not significantly change θ_0, then this linearization is a good approximation. By leveraging this linearization, we can obtain a closed-form solution for Eq.(12) as:

$$\begin{aligned}\theta^*(X_h) = (\triangledown_{\theta} f_{\theta_0}(X_h)^\top \triangledown_{\theta}\, f_{\theta_0}(X_h) + \beta I)^{-1} \\ \triangledown_{\theta}\, f_{\theta_0}(X_h)^\top (y_h - f_{\theta_0}(X_h)) + \theta_0.\end{aligned} \tag{16}$$

Building on this result, we can compute the bidirectional learning loss as:

$$\begin{aligned}\mathcal{L}_{bi}(X_h) = \frac{1}{2}(\|y_h - K_{X_h X_l}(K_{X_l X_l} + \beta I)^{-1} \\ (y_l - f_{\theta_0}(X_l))\|^2 + \|y_l - \\ K_{X_l X_h}(K_{X_h X_h} + \beta I)^{-1}(y_h - f_{\theta_0}(X_h))\|^2)\,,\end{aligned} \tag{17}$$

where $K(X_i, X_j) = \triangledown_{\theta} f_{\theta_0}(X_i)^\top \triangledown_{\theta}\, f_{\theta_0}(X_j)$. Following (Dukler et al., 2022), we can also only linearize the last layer of the network for simplicity, which defines the following kernel,

$$K(X_i, X_j) = BERT(X_i)^\top BERT(X_j), \tag{18}$$

where $BERT(X)$ denotes the feature of the sequence X extracted by BERT. Its kernel nature makes this approach suitable for small-data tasks (Arora et al., 2020), especially in drug discovery with high labeling cost of DNA/proteins.

3.2. Adaptive-γ

The forward mapping and the backward mapping play different roles in the sequence optimization process: the forward

Algorithm 1 Bidirectional Learning for Offline Model-based Biological Sequence Design

```
Input: The static dataset $\mathcal{D} = (X_l, y_l)$, the predefined target score $y_h = 10$, # iterations $T$, the pre-trained biological LM parameterized by $\theta_0$, the auxiliary model $f_{aux}(\cdot)$, the regularization $\beta$.
  Initialize $X_0$ as the sequence with the highest score in $\mathcal{D}$
  for $\tau = 0$ to $T - 1$ do
    Leverage Adaptive-$\gamma$ in Sec 3.2 to update the balance $\gamma$ by Eq. (23)
    if Adapt learning rate then
      Leverage Adaptive-$\eta$ in Sec 3.3 to update the learning rate $\eta$ by Eq. (25)
    end if
    Optimize $X$ by minimizing the bidirectional learning loss $\mathcal{L}_{bi}(X_\tau, \gamma)$ in Eq. (19):
      $X_{\tau+1} = X_\tau - \eta OPT(\nabla_X \mathcal{L}_{bi}(X_\tau, \gamma))$
  end for
  Return $X_h^* = X_T$
```

mapping encourages the high-scoring sequence to search for a higher target score (exploitation) and the backward mapping serves as a constraint. Since different sequences require different degrees of constraint, we introduce an extra hyperparameter $\gamma \in [0, 1]$ to control the balance between the corresponding terms in the loss function:

$$\mathcal{L}_{bi}(X_h, \gamma) = \gamma \mathcal{L}_{l2h}(X_h) + (1-\gamma)\mathcal{L}_{h2l}(X_h)\,. \tag{19}$$

Thus $\gamma = 1.0$ corresponds to the forward mapping alone, $\gamma = 0$ results in backward mapping, and $\gamma = 0.5$ leads to the bidirectional loss of (Chen et al., 2022c).

It is non-trivial to determine the most suitable value for γ since we do not know the ground-truth score for a new design. One possible solution is to train an auxiliary $f_{aux}(\cdot)$ to serve as a proxy evaluation. A reasonable auxiliary is a simple regression model fitted to the offline dataset. Although this auxiliary model cannot yield ground-truth scores, it can provide weak supervision signals to update γ, since the auxiliary model and the bidirectional learning provide complementary information. This is similar to co-teaching (Han et al., 2018) where two models leverage each other's view.

Formally, we introduce the *Adaptive-*γ framework. Given a good choice of γ, the produced $\boldsymbol{X}_h$ is expected to have a high score $f_{aux}(\boldsymbol{X}_h)$, based on which we can choose γ. To make the search for γ more efficient, we can formulate this process as a bi-level optimization problem:

$$\gamma^* = \arg\max_{\gamma} f_{aux}(\boldsymbol{X}_h^*(\gamma)), \tag{20}$$

$$\text{s.t.} \quad \boldsymbol{X}_h^*(\gamma) = \arg\min_{\boldsymbol{X}_h} \mathcal{L}_{bi}(\boldsymbol{X}_h, \gamma). \tag{21}$$

We can then use the hyper-gradient $\frac{\partial f_{aux}(\boldsymbol{X}_h^*(\gamma))}{\partial \gamma}$ to update γ. Specifically, the inner level solution can be approximated via a gradient descent step with a learning rate η:

$$\boldsymbol{X}_h^*(\gamma) = \boldsymbol{X}_h - \eta \frac{d\mathcal{L}_{bi}(\boldsymbol{X}_h, \gamma)}{d\boldsymbol{X}_h^\top}. \tag{22}$$

For the outer level, we update γ by hyper-gradient ascent:

$$\begin{aligned}\gamma &= \gamma + \eta' \frac{df_{aux}(\boldsymbol{X}_h^*(\gamma))}{d\gamma} = \gamma + \eta' \frac{df_{aux}(\boldsymbol{X}_h)}{d\boldsymbol{x}_h} \frac{d\boldsymbol{x}_h^*(\gamma)}{d\gamma} \\ &= \gamma + \eta' \eta \frac{df_{aux}(\boldsymbol{X}_h)}{d\boldsymbol{x}_h} \frac{d\mathcal{L}_{h2l}(\boldsymbol{X}_h) - \mathcal{L}_{l2h}(\boldsymbol{X}_h)}{d\boldsymbol{x}_h^\top},\end{aligned} \tag{23}$$

where we unroll the matrix form $\boldsymbol{X}_h$ as a vector form $\boldsymbol{x}_h$ for better illustration.

3.3. Adaptive-η

We now extend the *Adaptive-*γ framework to *Adaptive-*η. As the first learning rate adaptation module for offline model-based optimization, *Adaptive-*η is compatible with all gradient-based algorithms and can effectively finetune the learning rate η via the auxiliary model's weak supervision signal. All gradient-based methods that maximize $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{X})$ with respect to $\boldsymbol{X}$ have the following general form:

$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t + \eta OPT(\nabla_{\boldsymbol{X}} \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{X})|_{\boldsymbol{X}=\boldsymbol{X}_t}), \quad \text{for } t \in [0, \text{T}-1], \tag{24}$$

where η represents the learning rate of the optimizer. For methods such as simple gradient ascent (Grad), COMs (Trabucco et al., 2021), ROMA (Yu et al., 2021) and NEMO (Fu & Levine, 2021), $\mathcal{L}_{\boldsymbol{\theta}}(\cdot)$ is related to the proxy model $f_{\boldsymbol{\theta}}(\cdot)$; for BDI (Chen et al., 2022c) and our proposed method, BIB, $\mathcal{L}_{\boldsymbol{\theta}}(\cdot)$ is the negative of the bidirectional learning loss, i.e., $\mathcal{L}_{\boldsymbol{\theta}} = -\mathcal{L}_{bi}$.

Though the learning rate η can be adapted in some optimizers such as Adam (Kingma & Ba, 2015), these adaptations rely on only the past optimization history and do not consider the weak supervision signal from the auxiliary model. Our *Adaptive-*η optimizes η by solving:

$$\eta^* = \arg\max_{\eta} f_{aux}(\boldsymbol{X}_h^*(\eta)), \tag{25}$$

where η can be updated via gradient ascent methods. Considering the sequence optimization procedure is highly sensitive to the learning rate η, we reset η to η_0 at each iteration and update η from η_0,

$$\eta = \eta_0 - \eta' \frac{df_{aux}(\boldsymbol{X}_h^*(\eta))}{d\eta}. \tag{26}$$

In general, this stabilizes the optimization procedure.

4. Experiments

We conduct extensive experiments on DNA and protein design tasks, and aim to answer three research questions: (1) How does BIB compare with state-of-the-art algorithms? (2) Is every design component necessary in BIB? (3) Does the *Adaptive-*η module improve gradient-based methods?

4.1. Benchmark

We conduct experiments on two DNA tasks: TFBind8(r) and TFBind10(r), following (Chen et al., 2022c) and three protein tasks: avGFP, AAV and E4B, in (Ren et al., 2022) which have the most data points. See Appendix 7.3 for more details on task definitions and oracle evaluations. We further explore the performance of our method in relation to varying task dataset sizes and its computational cost in Appendix 7.4 and Appendix 7.5, respectively.

Following (Trabucco et al., 2021), we select the top $N = 128$ most promising sequences for each comparison method. Among these sequences, we report the maximum normalized ground truth score as the evaluation metric following (Ren et al., 2022).

4.2. Comparison Methods

We compare BIB with two groups of baselines: the gradient-based methods and the non-gradient-based methods. For a fair comparison, the pre-trained LM is used for all methods involving a proxy and we don't finetune the LM. The gradient-based methods include: 1) Grad: gradient ascent on existing sequences to obtain new sequences; 2) COMs (Trabucco et al., 2021): lower bounds the DNN model by the ground-truth values and then applies gradient ascent; 3) ROMA (Yu et al., 2021): incorporates a smoothness prior into the DNN model before gradient ascent steps; 4) NEMO (Fu & Levine, 2021): leverages the normalized maximum-likelihood estimator to bound the distance between the DNN model and the ground-truth values; 5) BDI (Chen et al., 2022c): adopts the infinitely wide NN and its NTK to compute bidirectional learning loss.

The non-gradient-based methods include: 1) BO-qEI (Wilson et al., 2017): builds an acquisition function for sequence exploration; 2) CMA-ES (Hansen, 2006): estimates the covariance matrix to adjust the sequence distribution towards the high-scoring region; 3) AdaLead (Sinai et al., 2020): performs a hill-climbing search on the proxy and then queries the sequences with high predictions; 4) CbAS (Brookes et al., 2019): builds a generative model for sequences above a property threshold and gradually adapts the distribution by increasing the threshold; 5) PEX (Ren et al., 2022): prioritizes the evolutionary search for protein sequences with low mutation counts; 6) GENH (Chan et al., 2021): enhances the score through a learned latent space.

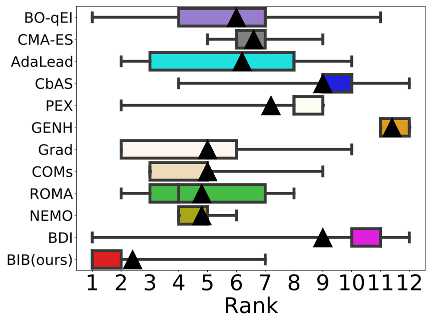


Figure 2. Rank minima and maxima are represented by whiskers; vertical lines and black triangles denote medians and means.

4.3. Training Details

We follow the training setting in (Chen et al., 2022c) if not specified. We choose OPT as the Adam optimizer (Kingma & Ba, 2015) for all gradient-based methods. We implement the auxiliary model as a linear layer with the feature from the pre-trained LM. We set the number of iterations T as 25 for all experiments following (Norn et al., 2021) and η_0 as 0.1 following (Chen et al., 2022c). We run every setting over 16 trials and report the mean and standard deviation. See Appendix 7.6 for other training details.

4.4. Results and Analysis

We report all experimental results in Table 1 and plot the ranking statistics in Figure 2. We make the following observations. (1) As shown in Table 1, BIB consistently outperforms the Grad method on all tasks, which demonstrates that our BIB can effectively mitigate the out-of-distribution issue. (2) Furthermore, BIB outperforms BDI on 4 out of 5 tasks, which demonstrates the effectiveness of the pre-trained biological LM over NTK. We have conducted experiments, reported in Appendix 7.7, to verify that the ranking of prediction performance is NN $>$ linearized pre-trained LM $>$ NTK. The reason why BDI outperforms BIB on TFBind10(r) may be that short sequences do not rely much on the rich sequential information from the pre-trained LM. (3) As shown in Figure 2, the gradient-based methods generally perform better than the non-gradient-based methods, as also observed by (Trabucco et al., 2021). (4) The gradient-based methods are inferior for the AAV task. One possible reason is that the design space of AAV (20^{28}) is much smaller than those of avGFP (20^{239}) and E4B (20^{102}), which makes the generative modeling and evolutionary algorithms more suitable. (5) This conjecture is also supported by the experimental results on two DNA design tasks. We compute the average ranking of gradient-based methods and non-grad-based methods on TFBind10(r) as 3.5 and 9.5, respectively, and the average ranking of gradient-based methods and non-grad-based methods on TFBind8(r) as 5.8 and 6.8, respectively. The advantage of gradient-based methods are larger ($9.5-3.5=6.0$) in TFBind10(r) than that ($6.8-5.8=1.0$) in TFBind8(r). (6) The generative modeling methods CbAS and GENH yield poor results on all tasks, probably because the high-dimensional data distribution is very hard to model. (7) Overall, BIB attains the best performance in 3 out of 5 tasks and achieves the best ranking results as shown in Table 1 and Figure 2.

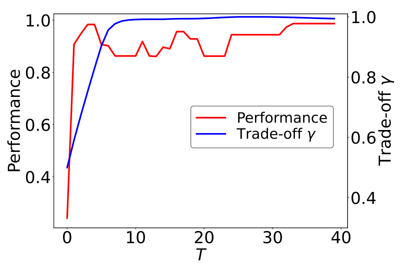
We also visualize the trend of performance (the maximum normalized ground truth score) and trade-off γ as a function of T on TFBind8(r) in Figure 3(a) and avGFP in Figure 3(b). The performance generally increases with the time step T and then stabilizes, which demonstrates the effectiveness and robustness of BIB. Furthermore, we find that the γ values of TFBind8(r) and avGFP generally increase at first. This means that BIB reduces the impact of the constraint to encourage a more aggressive search for a high target value during the initial phase. Then γ of TFBind8(r) continues to increase while the γ of avGFP decreases. We conjecture that the difference is caused by the sequence length. Small mutations of a biological sequence are enough to yield a good candidate (Ren et al., 2022). For the length-239 protein in avGFP, dramatic mutations 1) are not necessary and 2) can easily lead to out-of-distribution points. The weak supervision signal from the auxiliary model therefore encourages a tighter constraint towards the static dataset. By contrast, the DNA sequence is relatively short and a more widespread search of the sequence space can yield better results. To investigate this conjecture, we further visualize the trend of E4B in Figure 3(c). E4B also has long sequences (102) and we can observe its similar first-increase-then-decrease trend, although it is not as pronounced.

4.5. Ablation Studies

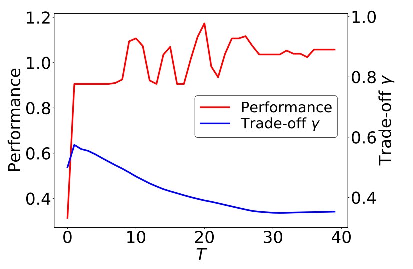
In this subsection, we conduct ablation studies to verify the effectiveness of the forward mapping, the backward mapping, and the *Adaptive*-γ module of BIB. We stress that forward mapping and backward mapping are not our contributions. In this paper, we propose deep linearization for

Table 1. Experimental results (maximum normalized ground truth score) for comparison.

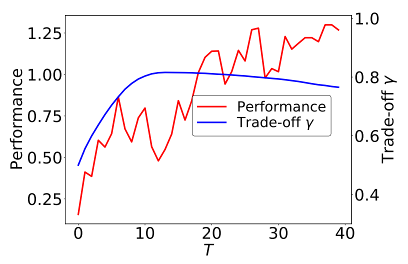
Method	TFBind8(r)	TFBind10(r)	avGFP	AAV	E4B	Rank Mean	Rank Median
$\mathcal{D}$(**best**)	0.242	0.248	0.314	0.452	0.224		
BO-qEI	0.940 ± 0.032	0.595 ± 0.028	0.888 ± 0.015	$\mathbf{0.591 \pm 0.002}$	0.436 ± 0.004	6.0/12	7.0/12
CMA-ES	0.930 ± 0.034	0.617 ± 0.031	0.909 ± 0.004	0.470 ± 0.006	0.748 ± 0.009	6.6/12	6.0/12
AdaLead	$\underline{0.941} \pm \underline{0.032}$	0.602 ± 0.028	0.885 ± 0.016	0.581 ± 0.002	0.433 ± 0.003	6.2/12	8.0/12
CbAS	0.878 ± 0.049	0.610 ± 0.035	0.785 ± 0.057	0.543 ± 0.002	0.349 ± 0.003	9.0/12	10.0/12
PEX	0.924 ± 0.041	0.612 ± 0.026	0.874 ± 0.033	$\underline{0.588} \pm \underline{0.002}$	0.397 ± 0.004	7.2/12	8.0/12
GENH	0.323 ± 0.000	0.448 ± 0.000	0.793 ± 0.000	0.452 ± 0.000	0.228 ± 0.000	11.4/12	11.0/12
Grad	$\underline{0.941} \pm \underline{0.026}$	0.630 ± 0.029	0.913 ± 0.027	0.463 ± 0.005	$\underline{1.219} \pm \underline{0.061}$	5.0/12	5.0/12
COMs	0.921 ± 0.039	0.637 ± 0.065	0.938 ± 0.048	0.511 ± 0.005	0.829 ± 0.026	5.0/12	5.0/12
ROMA	0.926 ± 0.032	0.634 ± 0.061	$\underline{0.975} \pm \underline{0.133}$	0.471 ± 0.005	1.198 ± 0.042	4.8/12	4.0/12
NEMO	0.930 ± 0.038	0.632 ± 0.024	0.914 ± 0.026	0.505 ± 0.005	1.036 ± 0.046	4.8/12	5.0/12
BDI	0.823 ± 0.000	$\mathbf{0.678 \pm 0.000}$	0.873 ± 0.000	0.452 ± 0.000	0.224 ± 0.000	9.0/12	11.0/12
BIB$_{\text{(ours)}}$	$\mathbf{0.952 \pm 0.033}$	$\underline{0.639} \pm \underline{0.032}$	$\mathbf{1.060 \pm 0.016}$	0.501 ± 0.007	$\mathbf{1.255 \pm 0.029}$	2.4/12	1.0/12



(a) TFBind8(r).



(b) avGFP.



(c) E4B.

Figure 3. Trend of performance and trade-off γ as a function of T.

Table 2. Ablation studies on BIB components.

Task	$\gamma = 0.0$	$\gamma = 1.0$	$\gamma = 0.5$	$\gamma = 0.5$ + Joint	BIB	BIB + Ada-η
TFB8(r)	0.936	0.933	0.947	0.935	0.952	0.956
TFB10(r)	0.611	0.637	0.616	0.622	0.639	0.639
avGFP	0.920	0.966	1.018	1.006	1.060	1.082
AAV	0.449	0.458	0.480	0.420	0.501	0.525
E4B	0.778	0.903	1.198	1.176	1.255	1.301

bidirectional mappings. This section explores whether both mappings remain effective after deep linearization has been performed. We report the experimental results in Table 2.

Forward mapping & Backward mapping. We can observe that bidirectional learning ($\gamma = 0.5$) performs better than both forward mapping ($\gamma = 1.0$) and backward mapping ($\gamma = 0.0$) alone in most tasks, which demonstrates the effectiveness of forward mapping and backward mapping. The advantage of *bidirectional mappings over the forward mapping* is larger in the long-sequence tasks like avGFP (238) and E4B (102) compared with the short-sequence tasks. A possible explanation is that the constraint is more important for long sequence tasks than short sequence design since the search space is large and many mutations can easily go out of distribution.

Adaptive-γ. BIB learns γ and this leads to improvements over bidirectional mappings ($\gamma = 0.5$) for all tasks, verifying the effectiveness of Adaptive-γ. We also consider the following variant,

$$\boldsymbol{X}^* = \arg\min_{\boldsymbol{X}_h} \mathcal{L}_{bi}(\boldsymbol{X}_h, 0.5) - f_{aux}(\boldsymbol{X}_h)\,, \tag{27}$$

which jointly optimizes the bidirectional learning loss $\mathcal{L}_{bi}(\boldsymbol{X}_h, 0.5)$ and the auxiliary term $f_{aux}(\boldsymbol{X}_h)$. We found this yields similar or even worse results than pure bidirectional learning. The reason may be that the weak supervision signal from $f_{aux}(\boldsymbol{X}_h)$ can serve as a guide to update the scalar γ but not as a component of the main optimization objective that directly updates the sequence.

In the final column of Table 2, we examine the Adaptive-η module. Adding this module leads to improvements on all five tasks, which demonstrates its effectiveness.

4.6. Adaptive-η

In this subsection, we aim to further demonstrate the effectiveness of the Adaptive-η module on all six gradient-based methods. We conduct experiments on two tasks: TFBind8(r) and avGFP, concurrently undertaking investigations on tasks within a broader domain to establish our method's broader applicability, as detailed in Appendix 7.8. Since the use of the infinitely wide neural network leads to poor performance for BDI, we modify its implementation via deep linearization so that it can make use of the pre-trained LM.

As shown in Table 3, *Adaptive-η* provides a consistent gain for all scenarios, which demonstrates the widespread applicability and effectiveness of the module. Furthermore, *Adaptive-η* leads to a maximum improvement of 1.4% in TFBind8(r) and 12.5% in avGFP. ROMA is the algorithm that benefits the most. One possible explanation is that ROMA incorporates a local smoothness prior that leads to more stable gradients, with which *Adaptive-η* can be more effective. Similar to Sec 4.5, we consider the variant,

$$\boldsymbol{X}^* = \arg\max_{\boldsymbol{X}_h} \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{X}_h) + f_{aux}(\boldsymbol{X}_h)\,, \tag{28}$$

which performs joint optimization instead of bi-level optimization on two objectives. As shown in Table 3, joint optimization generally deteriorates the performance. This again verifies that the auxiliary model can only serve as a guide instead of contributing to the main objective.

5. Related Work

Biological sequence design. There has been a wide range of algorithms for biological sequence design. Evolutionary algorithms (Sinai et al., 2020; Ren et al., 2022) leverage the learned surrogate model to provide evolution guidance towards the high-scoring region. (Angermueller et al., 2019) propose a flexible reinforcement learning framework where sequence design is a sequential decision-making problem. Bayesian optimization methods propose candidate solutions via an acquisition function (Terayama et al., 2021). Deep generative model methods design sequences in the latent space (Chan et al., 2021) or gradually adapt the distribution towards the high-scoring region (Brookes et al., 2019). GFlowNets (Jain et al., 2022) amortize the cost of search over learning and encourage diversity. Gradient-based methods leverage a surrogate model and its gradient information to maximize the desired property (Chen et al., 2022c; Norn et al., 2021; Tischer et al., 2020; Linder & Seelig, 2020). Our proposed BIB belongs to the last category and leverages the rich biophysical information (Ji et al., 2021; Elnaggar et al., 2021) to directly optimize the biological sequence.

Offline model-based optimization. A majority of sequence design algorithms (Angermueller et al., 2019; Sinai et al., 2020; Ren et al., 2022) focus on the online setting where wet-lab experimental results in the current round are analyzed to propose candidates in the next round. The problem of this setting is that wet-lab experiments are often very expensive, and thus a pure data-driven, offline approach is attractive and has received substantial research attention recently (Trabucco et al., 2022; Kolli et al., 2022; Beckham et al., 2022). Gradient-based methods have proven to be effective (Trabucco et al., 2021; Yu et al., 2021; Fu & Levine, 2021; Chen et al., 2022c). Among these algorithms, (Chen et al., 2022c) propose bidirectional mappings to distill information from the static dataset into a high-scoring design, which achieves state-of-the-art performances on a variety of tasks. However, this bidirectional learning is designed for general tasks, like robot and material design, and the rich biophysical information in millions of biological sequences is ignored. In this paper, we leverage recent advances in deep linearization to incorporate the rich biophysical information into bidirectional learning.

Bi-level optimization for hyperparameter optimization. Gradient-based bi-level optimization (Liu et al., 2021; Chen et al., 2022a; Giovannelli et al., 2021; Choe et al., 2022) has been widely used in hyperparameter optimization (Maclaurin et al., 2015; Franceschi et al., 2017; Pedregosa, 2016; Chen et al., 2022b; Li et al., 2022; Lorraine et al., 2020; Chen et al., 2021; Micaelli & Storkey, 2021; Zhong et al., 2022; Chi et al., 2022; Bohdal et al., 2021; Baydin et al., 2017). As shown in the *model training* scenario of Table 4, the inner level optimizes model parameters by minimizing the training loss, and the outer level optimizes hyperparameters by minimizing the validation loss. More specifically, (Maclaurin et al., 2015) exactly reverse the optimization dynamics to compute hyperparameter gradients. (Luketina et al., 2016) locally adjust hyperparameters to minimize validation loss. (Franceschi et al., 2018) unify hyperparameter optimization and meta-learning via bi-level optimization. (Donini et al., 2019) use past optimization information to simulate future behavior and compute the hypergradients efficiently. All of these previous works aim to improve model training and belong to the *model training* scenario in Table 4. By contrast, our Adaptive-γ and Adaptive-η belong to the second scenario: *design optimization*.

In our setting of offline model-based optimization, there is no validation set to provide hypergradient information to update hyperparameters, including the trade-off parameter γ and the learning rate η. Instead, inspired by previous work (Angermueller et al., 2019; Trabucco et al., 2021; Chan et al., 2021) that uses an auxiliary model to select candidates, we use the auxiliary to provide a weak supervision signal for hyperparameter optimization. This leads to a bi-level optimization task: the inner level optimizes design parameters by minimizing the training loss (bidirectional learning loss), and the outer level optimizes hyperparameters by maximizing the auxiliary score. As we can see, our formulation is different from that of previous work and thus can not compare with them directly.

6. Conclusion

In this paper, we propose bidirectional learning for offline model-based biological sequence. Our work is built on the recently proposed bidirectional learning approach (Chen et al., 2022c), which is designed for general inputs and relies on the NTK of an infinitely wide network to yield a closed-

Table 3. Adaptive-η on all gradient-based methods.

Method	TFBind8(r)						avGFP					
	Grad	COMs	ROMA	NEMO	BDI	BIB	Grad	COMs	ROMA	NEMO	BDI	BIB
Normal	0.941	0.921	0.926	0.930	0.947	0.952	0.913	0.938	0.975	0.914	1.018	1.060
Joint	0.941	0.921	0.931	0.932	0.935	0.925	0.913	0.905	0.923	0.906	1.006	1.009
Gain	0.000	0.000	0.005	0.002	−0.008	−0.027	0.000	−0.033	−0.052	−0.008	−0.012	−0.051
Ada-η	0.941	0.928	0.939	0.935	0.951	0.956	0.916	0.952	0.998	0.920	1.024	1.082
Gain	0.000	0.007	0.013	0.005	0.004	0.004	0.003	0.014	0.023	0.006	0.006	0.022

Table 4. Hyperparameter optimization from a gradient-based bi-level optimization view.

Scenario	Inner variables	Inner Objective	Outer variables	Outer Objective
model training	model params	minimize training loss (e.g. cross-entropy loss)	hyperparams	minimize validation loss
design optimization	design params	minimize training loss (e.g. bidirectional learning loss)	hyperparams	maximize auxiliary score

form loss computation. Though effective, the NTK cannot learn features. We build a proxy model using the pre-trained LM model with a linear head and apply the deep linearization scheme to the proxy, which can yield a closed-form loss and incorporate the wealth of biophysical information at the same time. In addition, we propose *Adaptive-γ* to maintain a proper balance between the forward mapping and the backward mapping by leveraging a weak supervision signal from an auxiliary model. Based on this framework, we further propose *Adaptive-η*, the first learning rate adaptation strategy compatible with all gradient-based offline model-based algorithms. Experimental results on DNA and protein sequence design tasks verify the effectiveness of BIB and *Adaptive-η*. In forthcoming research, our intention is to leverage the capabilities of more advanced, pre-trained protein models (Zhang et al., 2023a; 2022; 2023b; Chen et al., 2023) to further refine and optimize the process of protein design. We discuss potential negative impacts in Appendix 7.9, reproducibility in Appendix 7.10 and limitations of our research work in Appendix 7.11.

References

Achille, A., Golatkar, A., Ravichandran, A., Polito, M., and Soatto, S. Lqf: linear quadratic fine-tuning. In *Proc. Comp. Vision. Pattern. Rec.(CVPR)*, 2021.

Angermueller, C., Dohan, D., Belanger, D., Deshpande, R., Murphy, K., and Colwell, L. Model-based reinforcement learning for biological sequence design. In *Proc. Int. Conf. Learning Rep. (ICLR)*, 2019.

Arora, S., Du, S. S., Li, Z., Salakhutdinov, R., Wang, R., and Yu, D. Harnessing the power of infinitely wide deep nets on small-data tasks. In *Proc. Int. Conf. Learning Rep. (ICLR)*, 2020.

Bai, Y., Krause, B., Wang, H., Xiong, C., and Socher, R. Taylorized training: Towards better approximation of neural network training at finite width. *arXiv preprint arXiv:2002.04010*, 2020.

Barrera, L. A. et al. Survey of variation in human transcription factors reveals prevalent DNA binding changes. *Science*, 2016.

Baydin, A. G., Cornish, R., Rubio, D. M., Schmidt, M., and Wood, F. Online learning rate adaptation with hypergradient descent. *arXiv preprint arXiv:1703.04782*, 2017.

Beckham, C., Piche, A., Vazquez, D., and Pal, C. Towards good validation metrics for generative models in offline model-based optimisation. *arXiv preprint arXiv:2211.10747*, 2022.

Bohdal, O., Yang, Y., and Hospedales, T. Evograd: Efficient gradient-based meta-learning and hyperparameter optimization. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

Brookes, D., Park, H., and Listgarten, J. Conditioning by adaptive sampling for robust design. In *Proc. Int. Conf. Machine Learning (ICML)*, 2019.

Bryant, D. H., Bashir, A., Sinai, S., Jain, N. K., Ogden, P. J., Riley, P. F., Church, G. M., Colwell, L. J., and Kelsic, E. D. Deep diversification of an AAV capsid protein by machine learning. *Nature Biotechnology*, 2021.

Chan, A., Madani, A., Krause, B., and Naik, N. Deep extrapolation for attribute-enhanced generation. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

Chen, C., Zheng, S., Chen, X., Dong, E., Liu, X. S., Liu, H., and Dou, D. Generalized data weighting via class-level gradient manipulation. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

Chen, C., Chen, X., Ma, C., Liu, Z., and Liu, X. Gradient-based bi-level optimization for deep learning: A survey. *arXiv preprint arXiv:2207.11719*, 2022a.

Chen, C., Ma, C., Chen, X., Song, S., Liu, H., and Liu, X. Unbiased implicit feedback via bi-level optimization. *arXiv preprint arXiv:2206.00147*, 2022b.

Chen, C., Zhang, Y., Fu, J., Liu, X., and Coates, M. Bidirectional learning for offline infinite-width model-based optimization. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2022c.

Chen, C., Zhou, J., Wang, F., Liu, X., and Dou, D. Structure-aware protein self-supervised learning. *Bioinformatics*, 2023.

Chi, Z., Gu, L., Liu, H., Wang, Y., Yu, Y., and Tang, J. Metafscil: a meta-learning approach for few-shot class incremental learning. In *CVPR*, 2022.

Choe, S. K., Neiswanger, W., Xie, P., and Xing, E. Betty: An automatic differentiation library for multilevel optimization. *arXiv preprint arXiv:2207.02849*, 2022.

Donini, M., Franceschi, L., Pontil, M., Majumder, O., and Frasconi, P. Marthe: Scheduling the learning rate via online hypergradients. *arXiv preprint arXiv:1910.08525*, 2019.

Dukler, Y., Achille, A., Paolini, G., Ravichandran, A., Polito, M., and Soatto, S. DIVA: Dataset derivative of a learning task. In *Proc. Int. Conf. Learning Rep. (ICLR)*, 2022.

Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., et al. ProtTrans: towards cracking the language of lifes code through self-supervised deep learning and high performance computing. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2021.

Franceschi, L., Donini, M., Frasconi, P., and Pontil, M. Forward and reverse gradient-based hyperparameter optimization. In *Proc. Int. Conf. Machine Learning (ICML)*, 2017.

Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. In *Proc. Int. Conf. Machine Learning (ICML)*, 2018.

Fu, J. and Levine, S. Offline model-based optimization via normalized maximum likelihood estimation. *Proc. Int. Conf. Learning Rep. (ICLR)*, 2021.

Giovannelli, T., Kent, G., and Vicente, L. N. Inexact bilevel stochastic gradient methods for constrained and unconstrained lower-level problems. *arXiv preprint arXiv:2110.00604*, 2021.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. Co-teaching: robust training of deep neural networks with extremely noisy labels. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2018.

Hansen, N. The CMA evolution strategy: a comparing review. *Towards A New Evolutionary Computation*, 2006.

Jain, M., Bengio, E., Hernandez-Garcia, A., Rector-Brooks, J., Dossou, B. F., Ekbote, C. A., Fu, J., Zhang, T., Kilgour, M., Zhang, D., et al. Biological sequence design with GFlowNets. In *Proc. Int. Conf. Machine Learning (ICML)*, 2022.

Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. DNABERT: pre-trained bidirectional encoder representations from transformers model for DNA-language in genome. *Bioinformatics*, 2021.

Killoran, N., Lee, L. J., Delong, A., Duvenaud, D., and Frey, B. J. Generating and designing DNA with deep generative models. *arXiv preprint arXiv:1712.06148*, 2017.

Kingma, D. P. and Ba, J. Adam: a method for stochastic optimization. In *Proc. Int. Conf. Learning Rep. (ICLR)*, 2015.

Kolli, S., Lu, A. X., Geng, X., Kumar, A., and Levine, S. Data-driven optimization for protein design: workflows, algorithms and metrics. In *ICLR2022 Machine Learning for Drug Discovery*, 2022.

Kumar, V., Liu, H., and Wu, C. Drug repurposing against SARS-CoV-2 receptor binding domain using ensemble-based virtual screening and molecular dynamics simulations. *Computers in Biology and Medicine*, 2021.

Lee, Y. and Yu, H. ProtFIM: Fill-in-middle protein sequence design via protein language models, 2023. URL `https://openreview.net/forum?id=9XAZBUfnefS`.

Li, G., Togo, R., Ogawa, T., and Haseyama, M. Compressed gastric image generation based on soft-label dataset distillation for medical data sharing. *Computer Methods and Programs in Biomedicine*, 2022.

Linder, J. and Seelig, G. Fast differentiable DNA and protein sequence optimization for molecular design. *arXiv preprint arXiv:2005.11275*, 2020.

Linder, J. and Seelig, G. Fast activation maximization for molecular sequence design. *BMC Bioinformatics*, 2021.

Liu, R., Gao, J., Zhang, J., Meng, D., and Lin, Z. Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Lorraine, J., Vicol, P., and Duvenaud, D. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Luketina, J., Berglund, M., Greff, K., and Raiko, T. Scalable gradient-based tuning of continuous regularization hyperparameters. In *Proc. Int. Conf. Machine Learning (ICML)*, 2016.

Maclaurin, D., Duvenaud, D., and Adams, R. Gradient-based hyperparameter optimization through reversible learning. In *Proc. Int. Conf. Machine Learning (ICML)*, 2015.

Micaelli, P. and Storkey, A. J. Gradient-based hyperparameter optimization over long horizons. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

Mu, F., Liang, Y., and Li, Y. Gradients as features for deep representation learning. *Proc. Int. Conf. Learning Rep. (ICLR)*, 2020.

Murray, C. J., Ikuta, K. S., Sharara, F., Swetschinski, L., Aguilar, G. R., Gray, A., Han, C., Bisignano, C., Rao, P., Wool, E., et al. Global burden of bacterial antimicrobial resistance in 2019: a systematic analysis. *The Lancet*, 2022.

Norn, C., Wicky, B. I., Juergens, D., Liu, S., Kim, D., Tischer, D., Koepnick, B., Anishchenko, I., Baker, D., and Ovchinnikov, S. Protein sequence design by conformational landscape optimization. *Proceedings of the National Academy of Sciences*, 2021.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: an imperative style, high-performance deep learning library. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2019.

Pedregosa, F. Hyperparameter optimization with approximate gradient. In *Proc. Int. Conf. Machine Learning (ICML)*, 2016.

Ren, Z., Li, J., Ding, F., Zhou, Y., Ma, J., and Peng, J. Proximal exploration for model-guided protein sequence design. *Proc. Int. Conf. Machine Learning (ICML)*, 2022.

Sarkisyan, K. S. et al. Local fitness landscape of the green fluorescent protein. *Nature*, 2016.

Sinai, S., Wang, R., Whatley, A., Slocum, S., Locane, E., and Kelsic, E. D. Adalead: a simple and robust adaptive greedy search algorithm for sequence design. *arXiv preprint arXiv:2010.02141*, 2020.

Starita, L. M., Pruneda, J. N., Lo, R. S., Fowler, D. M., Kim, H. J., Hiatt, J. B., Shendure, J., Brzovic, P. S., Fields, S., and Klevit, R. E. Activity-enhancing mutations in an E3 ubiquitin ligase identified by high-throughput mutagenesis. *Proceedings of the National Academy of Sciences*, 2013.

Terayama, K., Sumita, M., Tamura, R., and Tsuda, K. Black-box optimization for automated discovery. *Accounts of Chemical Research*, 2021.

Tischer, D., Lisanza, S., Wang, J., Dong, R., Anishchenko, I., Milles, L. F., Ovchinnikov, S., and Baker, D. Design of proteins presenting discontinuous functional sites using deep learning. *Biorxiv*, 2020.

Trabucco, B., Kumar, A., Geng, X., and Levine, S. Conservative objective models for effective offline model-based optimization. In *Proc. Int. Conf. Machine Learning (ICML)*, 2021.

Trabucco, B., Geng, X., Kumar, A., and Levine, S. Design-Bench: benchmarks for data-driven offline model-based optimization. *arXiv preprint arXiv:2202.08450*, 2022.

Wilson, J. T., Moriconi, R., Hutter, F., and Deisenroth, M. P. The reparameterization trick for acquisition functions. *arXiv preprint arXiv:1712.00424*, 2017.

Yang, G. and Hu, E. J. Tensor programs iv: feature learning in infinite-width neural networks. In *Proc. Int. Conf. Machine Learning (ICML)*, 2021.

Yu, S., Ahn, S., Song, L., and Shin, J. Roma: robust model adaptation for offline model-based optimization. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

Zhang, Z., Xu, M., Jamasb, A., Chenthamarakshan, V., Lozano, A., Das, P., and Tang, J. Protein representation learning by geometric structure pretraining. *arXiv preprint arXiv:2203.06125*, 2022.

Zhang, Z., Xu, M., Chenthamarakshan, V., Lozano, A., Das, P., and Tang, J. Enhancing protein language models with structure-based encoder and pre-training. *arXiv preprint arXiv:2303.06275*, 2023a.

Zhang, Z., Xu, M., Lozano, A., Chenthamarakshan, V., Das, P., and Tang, J. Physics-inspired protein encoder pre-training via siamese sequence-structure diffusion trajectory prediction. *arXiv preprint arXiv:2301.12068*, 2023b.

Zhong, T., Chi, Z., Gu, L., Wang, Y., Yu, Y., and Tang, J. Meta-dmoe: Adapting to domain shift by meta-distillation from mixture-of-experts. *arXiv preprint arXiv:2210.03885*, 2022.

7. Appendix

7.1. DNA Embedding

To incorporate richer contextual information, the DNA LM (Ji et al., 2021) adopts the k-mer sequence representation, which is widely used in DNA sequence analysis. For example, the sequence $ATGGCT$ has its 3-mer representation

Table 5. Experimental results on different pre-trained LMs for comparison.

Pre-trained LM	avGFP	AAV	E4B
ProtAlbert	0.907 ± 0.004	0.478 ± 0.004	0.552 ± 0.023
ProtBert$_{(\text{adopted})}$	1.060 ± 0.016	0.501 ± 0.007	1.255 ± 0.029
ProtBert-BFD	1.119 ± 0.116	0.549 ± 0.009	1.880 ± 0.054

as $\{ATG, TGG, GGC, GCT\}$. In this paper, we adopt its 3-mer representation and compute the probability of the 3-mer token by multiplying the probabilities of the three individual bases. The 3-mer representation is then sent to the pre-trained DNA LM.

7.2. Different Pretrained LMs

As shown in Table 5, we have tested the ProtBERT, ProtAlbert, and ProtBert-BFD models and found that better-quality models generally work better. The publicly available pre-trained DNA models are limited and thus we only perform experiments on the protein tasks. (Elnaggar et al., 2021) demonstrate that the language model performances follow the ordering: ProtBert-BFD > ProtBert > ProtAlbert. We can see that the performance ranks over the three protein tasks avGFP, AAV, and E4B are the same in Table 5.

7.3. Task Details

We conduct experiments on two DNA tasks following (Chen et al., 2022c) and three protein tasks in (Ren et al., 2022) which have the most data points. We report the dataset details in Table 6.

DNA Task 1 TFBind8(r). The goal is to find a length-8 DNA sequence to maximize the binding activity score with a particular transcription factor, $\mathrm{SIX6REFR1}$ (Barrera et al., 2016). We sample 5000 data points for the offline algorithms following (Chen et al., 2022c).

DNA Task 2 TFBind10(r). The task TFBind10(r) is the same as TFBind8(r) except that the goal is to find a length-10 DNA sequence. Both DNA tasks measure the entire search space and we adopt these measurements as the approximate ground-truth evaluation.

Protein Task 1 avGFP. This task aims to find a protein sequence with approximately 239 amino acids to maximize the fluorescence level of Green Fluorescent Proteins (Sarkisyan et al., 2016). The task oracle is constructed by using the full unobserved dataset (around 52,000 points) following (Ren et al., 2022). The oracle passes the average of the residue embeddings from the pre-trained Prot-T5 (Elnaggar et al., 2021) into a linear layer and then fits the dataset. The following two task oracles take the same form. Offline algorithms can only access the lowest-scoring 26,000 points.

Protein Task 2 AAV. The goal is to engineer a 28-amino acid segment (positions 561–588) of the VP1 protein to remain viable for gene therapy (Bryant et al., 2021). We use the entire $284,000$ data points to build the oracle and the lowest-scoring $142,000$ points for the offline algorithms.

Protein Task 3 E4B. This task aims to design a protein (around 102 amino acids) to maximize the ubiquitination rate to the target protein (Starita et al., 2013). The full dataset with around $100,000$ points is used to build the oracle and the bottom half is used for the offline algorithms.

(Lee & Yu, 2023) evaluate the sequence generation and structure conservation simultaneously with the help of Alphafold2, which provides a more accurate evaluation.

Oracle Parameterization. The parameterization of the oracle is different from that of the regression model from two aspects: 1) model architecture; 2) pre-trained information source. First, the oracle adopts the Prot-T5 model which consists of an encoder and a decoder, while the regression model adopts the Prot-BERT model which only has an encoder. Second, Prot-T5 is trained on the BFD and UniRef100 datasets and ProtBert is trained on the UniRef50 dataset. These two points demonstrate that the oracle and the regression model are different function classes. We choose the Prot-T5 model as the oracle because this is the state-of-the-art protein LM to extract features and recent work (Elnaggar et al., 2021) has demonstrated its effectiveness. In order to test how related the Prot-T5 (oracle)/Prot-BERT(proxy) models are, we trained them on a sampled training dataset and compared the test predictions of the testing set. By evaluating the Pearson correlation coefficient (PCC) between the two prediction errors PCC(ProtT5 predictions - test labels, ProtBERT predictions -test labels), we obtain 0.0104 on avGFP, -0.0005 on AAV, and -0.0062 on E4B. These results suggest that the two models are not strongly related in terms of the predictions they form.

Following (Trabucco et al., 2021), we select the top $N = 128$ most promising sequences for each comparison method. Among these sequences, we report the maximum normalized ground truth score as the evaluation metric following (Ren et al., 2022).

7.4. Different Dataset Size

We have tested the performance of BIB as a function of dataset size (N= 20, 40, 60, 80, 100) in all tasks. As shown in Table 7, the proposed approach BIB performs reasonably well even with a small initial dataset size of N=20/40, and its performance improves gradually as we increase N, which demonstrates the effectiveness and robustness of BIB.

Moreover, we have also included the performance of the ROMA method as a function of the initial dataset size in Table 8, which shows that the performances of ROMA are

Table 6. Dataset details.

Task	Metric	Min of $\mathcal{D}$	Max of $\mathcal{D}$	Min of $\mathcal{D}_{entire}$	Max of $\mathcal{D}_{entire}$
TFBind8(r)	binding activity	0.000	0.242	0.000	1.000
TFBind10(r)	binding activity	-1.859	-0.869	-1.859	2.129
avGFP	fluorescence level	1.283	2.175	1.283	4.123
AAV	viruses viability	-11.176	-1.814	-11.176	9.536
E4B	ubiquitination rate	-3.589	-0.770	-3.589	8.998

generally worse than our proposed BIB, highlighting the effectiveness of our approach.

7.5. Computational Cost

The computational time of BIB consists of three parts: the pre-computation, the Adaptive-γ, and the gradient update. All experiments are performed on one NVIDIA 32G V100 in the same cluster. We report the computational time of three parts in Table 9. The pre-computation results in a time complexity of O(N^3), where N is the size of the static dataset $\mathcal{D}$. It is worth noting that for datasets with larger sizes such as AAV, the pre-computation time of our method is much higher compared to smaller datasets such as TFBind8(r). However, the pre-computed $(\boldsymbol{K}_{\boldsymbol{X}_l\boldsymbol{X}_l}+\beta\boldsymbol{I})^{-1}(\boldsymbol{y}_l-f_{\boldsymbol{\theta}_0}(\boldsymbol{X}_l))$ can be used for every gradient update iteration in different trials. Since N is typically not very large in our experiments, the computational time is usually reasonable.

We further report the computational time of every iteration for all gradient-based offline MBO methods. As shown in Table 10, the computational time of all methods is at the same scale, with the exception of the high computation time of BIB on TFBind8(r) and TFBind10(r). We suspect that this is caused by its encoding strategy, as we explain in Appendix 7.1.

In this biological sequence design setting, the majority of the time in the production cycle will be spent on evaluating the unknown objective function. As a result, the time difference between the methods for obtaining the high score design is less significant in a real production environment.

7.6. Training Details

We use Pytorch (Paszke et al., 2019) to run all experiments on one V100 GPU. Following the setting in (Norn et al., 2021), we introduce a length-L protein sequence as a continuous random matrix $\boldsymbol{X}_h \in R^{L\times 20}$ ($\boldsymbol{X}_h \in R^{L\times 4}$ for DNA), initialized using a normal distribution with the mean 0 and the standard deviation of 0.01. To make this sequence correspond correctly to the candidate, we exchange the largest value in $\boldsymbol{X}[l,:]$ with the value in the amino acid index.

We set the initial learning rate as 0.1 for all tasks, following the practice in (Chen et al., 2022c). We conduct additional experiments with different learning rates (0.05 and 0.20) on two design tasks TFBind8(r) and avGFP using the baseline ROMA and our method. The results of TFBind8(r) and avGFP are reported in Table 11 and Table 12, respectively. While there is still a visible performance gap between different learning rates in the non-adaptive version, the use of our Adaptive-η module generally improves performance and greatly reduces this gap.

7.7. Ranking Performance

As for prediction performances, the rank should be: a NN $>$ linearized pre-trained LM $>$ NTK. We have conducted experiments to verify this. We sample half of the data, train a model to predict another half data, and report the mean squared loss here and in Table 13. A small mean squared loss indicates a good prediction performance; thus, we have verified the above ranking order.

7.8. Broader Applicability

We have conducted experiments on four additional design tasks in the design-bench (Trabucco et al., 2021), namely Superconductor, Ant Morphology, D'Kitty Morphology, and Hopper Controller. These tasks have different characteristics from the biological sequence design task, and thus provide a more diverse set of experiments to evaluate the effectiveness of the Adaptive-η module. The results, as shown in Table 14, Table 15, Table 16 and Table 17, demonstrate that the Adaptive-η module can improve the performance of gradient-based methods in a wider range of offline model-based optimization tasks. These findings underscore the broader applicability of our method within the realm of offline model-based optimization.

7.9. Negative Impact

Protein sequence design aims to find a protein sequence with a particular biological function, which has a broad application scope. This can lead to improved drugs that are highly beneficial to society. For instance, designing the antibody protein for SARS-COV-2 can potentially save millions of human lives (Kumar et al., 2021) and designing novel anti-microbial peptides (short protein sequences) is central to tackling the growing public health risks caused by anti-microbial resistance (Murray et al., 2022). Unfortunately,

Table 7. Experimental results of BIB as a function of the dataset size.

Dataset size	20	40	60	80	100
TFBind8(r)	0.849 ± 0.027	0.883 ± 0.036	0.890 ± 0.033	0.911 ± 0.042	0.923 ± 0.049
TFBind10(r)	0.248 ± 0.000	0.596 ± 0.035	0.602 ± 0.023	0.616 ± 0.024	0.632 ± 0.036
avGFP	0.902 ± 0.013	0.901 ± 0.009	0.905 ± 0.003	0.906 ± 0.000	0.909 ± 0.007
AAV	0.410 ± 0.003	0.399 ± 0.002	0.413 ± 0.002	0.456 ± 0.003	0.480 ± 0.009
E4B	0.574 ± 0.048	0.603 ± 0.034	0.853 ± 0.043	0.864 ± 0.022	0.997 ± 0.034

Table 8. Experimental results of ROMA as a function of the dataset size.

Dataset size	20	40	60	80	100
TFBind8(r)	0.783 ± 0.062	0.782 ± 0.051	0.869 ± 0.040	0.886 ± 0.029	0.891 ± 0.022
TFBind10(r)	0.545 ± 0.016	0.580 ± 0.012	0.592 ± 0.021	0.580 ± 0.004	0.598 ± 0.019
avGFP	0.905 ± 0.003	0.906 ± 0.004	0.906 ± 0.001	0.906 ± 0.002	0.906 ± 0.001
AAV	0.304 ± 0.024	0.360 ± 0.020	0.379 ± 0.022	0.410 ± 0.020	0.432 ± 0.029
E4B	0.255 ± 0.021	0.451 ± 0.025	0.693 ± 0.044	0.733 ± 0.032	0.696 ± 0.020

it is possible to direct the research results towards harmful purposes such as the design of biochemical weapons. As researchers, we believe that we must be aware of the potential harm of any research outcomes, and carefully consider whether the possible benefits outweigh the risks of harmful consequences. We also must recognize that we cannot control how the research may be used. In the case of this paper, we are confident that there is a much greater chance that the research outcomes will have a beneficial effect. We do not consider that there are any immediate ethical concerns with the research endeavour.

7.10. Reproducibility Statement

We provide the code implementation of BIB and *Adaptive-*η here and we also attach the code in the supplementary material. We describe DNA/protein benchmarks in Sec. 4.1 and training details in Sec. 4.3. We explain how to obtain the sequence embedding from the pre-trained LM and how to perform gradient ascent on the sequence in Sec. 2.

7.11. Limitations

We note that, due to the nature of offline optimization, we propose designs that are outside the available datasets. Evaluation of the performance of such designs must then rely on training an oracle model based on the entire available data. We are careful to ensure that (i) the oracle is significantly structurally different from the model(s) used within the optimization algorithm, (ii) the oracle is accurate, especially in terms of rank correlation; and (iii) oracle residuals and regression model residuals are uncorrelated. However, even after these measures are taken, there is a danger that the proposed designs are not biologically optimal, i.e., that the oracle performance is not genuinely reflective of performance that would be observed in experiments. This concern can only be alleviated via further biological experiments that examine how well off-line optimization algorithms perform in practice.

Table 9. Computational time of BIB on all tasks.

Method	TFBind8(r)	TFBind10(r)	avGFP	AAV	E4B
N	5000	5000	26000	142000	100000
Pre-computation (s)	1.05	0.95	6.02	2772.76	118.71
Ada-γ per iteration (s)	40.43	57.90	3.83	3.89	4.95
Grad-update per iteration (s)	44.15	52.13	4.55	4.01	3.90

Table 10. Gradient update time per iteration.

Method	TFBind8(r)	TFBind10(r)	avGFP	AAV	E4B
Grad (s)	6.48	7.81	4.01	5.06	3.76
COMs (s)	6.86	7.89	3.66	3.35	4.75
ROMA (s)	6.31	7.65	3.82	3.51	3.64
NEMO (s)	12.57	14.12	4.17	3.79	3.92
BDI (s)	10.23	10.99	13.97	31.96	27.61
BIB (s)	84.58	110.03	8.38	7.90	8.85

Table 11. Results for TFBind8(r)

Lr	0.05	0.10	0.20
ROMA	0.914	0.926	0.943
ROMA + Ada-eta	0.938	0.939	0.946
BIB	0.922	0.952	0.955
BIB + Ada-eta	0.947	0.956	0.956

Table 12. Results for avGFP

Lr	0.05	0.10	0.20
ROMA	0.941	0.975	0.978
ROMA + Ada-eta	0.972	0.998	0.998
BIB	0.999	1.060	1.015
BIB + Ada-eta	1.063	1.082	0.974

Table 13. Mean squared prediction losses for comparison.

Method	TFBind8(r)	TFBind10(r)	avGFP	AAV	E4B
Finetuned NN	0.101 ± 0.001	1.130 ± 0.041	0.323 ± 0.006	5.148 ± 0.074	0.683 ± 0.012
Linearized NN	0.107 ± 0.000	1.618 ± 0.000	3.956 ± 0.000	23.041 ± 0.000	1.050 ± 0.000
NTK	0.111 ± 0.000	1.840 ± 0.000	4.866 ± 0.000	24.451 ± 0.000	1.075 ± 0.000

Table 14. Results for Task 1 - Superconductor

Task	Grad	COMs	ROMA	NEMO	BDI
Normal	0.483	0.487	0.476	0.488	0.520
Ada	0.494	0.486	0.524	0.513	0.521
Gain	0.011	-0.001	0.048	0.025	0.001

Table 15. Results for Task 2 - Ant Morphology

Task	Grad	COMs	ROMA	NEMO	BDI
Normal	0.764	0.866	0.814	0.814	0.962
Ada	0.906	0.842	0.885	0.943	0.958
Gain	0.142	-0.024	0.071	0.129	-0.004

Table 16. Results for Task 3 - D'Kitty Morphology

Task	Grad	COMs	ROMA	NEMO	BDI
Normal	0.888	0.835	0.905	0.924	0.941
Ada	0.909	0.896	0.903	0.954	0.958
Gain	0.021	0.061	-0.002	0.030	0.017

Table 17. Results for Task 4 - HopperController

Task	Grad	COMs	ROMA	NEMO	BDI
Normal	0.989	1.224	1.849	1.959	1.989
Ada	1.060	1.111	1.901	1.950	2.012
Gain	0.071	-0.113	0.052	-0.009	0.023