Overconfident Oracles: Limitations of In Silico Sequence Design Benchmarking

Shikha Surana [1] **Nathan Grinsztajn** [1] **Timothy Atkinson** [1] **Paul Duckworth** [1] **Thomas D. Barrett** [1]

Abstract

Machine learning methods can automate the *in silico* design of biological sequences, aiming to reduce costs and accelerate medical research. Given the limited access to wet labs, *in silico* design methods commonly use an oracle model to evaluate *de novo* generated sequences. However, the use of different oracle models across methods makes it challenging to compare them reliably, motivating the question: are *in silico* sequence design benchmarks reliable? In this work, we examine 12 sequence design methods that utilise ML oracles common in the literature and find that there are significant challenges with their cross-consistency and reproducibility. Indeed, oracles differing by architecture, or even just training seed, are shown to yield conflicting relative performance with our analysis suggesting poor out-of-distribution generalisation as a key issue. To address these challenges, we propose supplementing the evaluation with a suite of biophysical measures to assess the viability of generated sequences and limit out-of-distribution sequences the oracle is required to score, thereby improving the robustness of the design procedure. Our work aims to highlight potential pitfalls in the current evaluation process and contribute to the development of robust benchmarks, ultimately driving the improvement of *in silico* design methods.

1. Introduction

Utilising generative machine learning (ML) to design biological sequences that maximise desired properties, such as binding affinity or expression level, is important for advancing the medical research field and its applications. Experiments conducted *in vitro* are expensive and time-consuming, and thus, leveraging ML to automate the *in silico* design of sequences to have a high likelihood of *in vitro* success can reduce costs and accelerate research progress.

In recent years, numerous *in silico* design methods have been proposed to generate biological sequences (Trabucco et al., 2021; Jain et al., 2022; Ren et al., 2022; Chen et al., 2023; Kim et al., 2024). Typically, biological sequences can be 100's to 1000's of characters long, leading to vast search spaces, but often have access only to a limited dataset of example sequences and their ground-truth values. As a result, to evaluate sequence design methods, an ML oracle model is often trained on the dataset and used to score *de novo* generated sequences, thereby simulating the wet lab evaluation. However, within the community there is a general lack of consensus on the specific oracle model parameters and architecture, which results in the use of different oracle models across different studies. It is common practice to propose novel sequence design methods alongside a tailored evaluation pipeline, including the choice of oracle. Inconsistency across oracle models hinders any reliable comparisons of design methods, and brings into question the robustness of *in silico* sequence design benchmarks.

Contributions Our first contribution is to investigate the reliability of 12 design methods. Specifically, whether superfluous changes to the oracle can impact the relative performance of methods and affect their overall ranking when scored against: (1) the same oracle architecture trained with five different seeds and (2) three different oracle architectures. We perform this on two different tasks: 5' untranslated region (UTR, DNA task) (Sample et al., 2019), and green fluorescence protein (GFP, protein task) (Sarkisyan et al., 2016). We demonstrate high variance and a lack of consensus in the methods' relative performance and present insights suggesting that issues arise from the oracle's poor out-of-distribution (OOD) generalisation.

Our second contribution introduces a suite of biophysical measures, specifically tailored for DNA and protein sequence design tasks, to assess the biological validity of *de novo* generated sequences. We demonstrate that these measures are critical due to ML oracle's i) poor OOD generalization, which necessitates reducing OOD sequences being evaluated and improving reliability, and ii) lack of biological knowledge, which prevents it from filtering out biologically unfit sequences.

[1] InstaDeep, London UK. Correspondence to: Shikha Surana <s.surana@instadeep.com>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

Our work complements the growing interest within the community towards improving *in silico* benchmarks for biological tasks. Recent studies have proposed biophysical measures to improve the benchmarks for *de novo* structure-based (Buttenschoen et al., 2024; Harris et al., 2023) and sequence-based (Frey et al., 2024; Spinner et al., 2024) design tasks. In our work, we highlight critical limitations of the current *in silico* protein and DNA sequence design benchmarks, and further, introduce additional biophysical measures to improve the robustness and reliability.

2. Related Work

Sequence Design Methods and Tasks Several works have developed offline methods to tackle the problem of sequence design, particularly through reinforcement learning (Angermueller et al., 2019), population-based optimisation (Angermueller et al., 2020), model-based optimisation (Trabucco et al., 2021; Chen et al., 2023), deep generative models (Kumar & Levine, 2020; Jain et al., 2022; Kim et al., 2024), and evolutionary search (Ren et al., 2022). To help provide a level of standardisation for biological sequence design tasks and methods, there has recently been several open-source resources, for tasks: ProteinGym (Notin et al., 2024), DesignBench (Trabucco et al., 2022), and FLEXS[1], and for methods: Design Baselines (Trabucco et al., 2021). Our experiments include some of the methods mentioned above, as well as Design Bench and Design Baselines suites.

Evaluation of *in silico* Benchmarks A recent, important area of research is assessing the physical and chemical plausibility of ML-generated solutions for biological tasks. Previous studies demonstrate that ML-based methods tend to generate physically implausible structures for tasks such as docking (Buttenschoen et al., 2024) and structure-based drug design (Harris et al., 2023), and these works present a suite of biophysical measures to validate the biological viability of generated complexes. Similarly, prior works have presented biologically-inspired measures for protein design tasks (Frey et al., 2024; Spinner et al., 2024). Our work expands the scope to include both protein and DNA tasks, and introduces additional measures for each task setting that are evaluated against both task and external datasets.

Generalisation of Sequence-Scoring Models The work by Tagasovska et al. (2024) discusses how surrogate models can fail to identify true casual and mechanistic links between (parts of) the sequences and the biological property of interest which is scored, leading to poor generalisation to unseen sequences. The findings of this study raise questions about the oracle's generalization ability and we investigate this in our work.

3. Experimental Setup

In this section we outline the biological sequence design datasets, tasks (including the oracle models), and the generators that will be used in the subsequent experiment sections.

3.1. Datasets

This work considers three datasets: green fluorescence protein (GFP), 5' untranslated region (UTR), and transcription factor binding sequences of length 8 (TFBind-8), each consisting of sequences annotated with a particular biological characteristic of interest. The distribution of ground-truth scores for each task is presented in Appendix Figure 4.

GFP dataset contains protein sequences of length 237, consisting of 20 possible amino acids, annotated with a ground-truth value corresponding to its fluorescence level. It is curated by Sarkisyan et al. (2016) and comprises $51,715$ unique sequences. Each sequence in this dataset has up to 15 mutational edits, with an average of 3.7 edits compared to the wild-type sequence.

UTR dataset contains DNA sequences of length 50, constructed using 4 nucleobases: adenine (A), guanine (G), cytosine (C), and thymine (T). It consists of $280,000$ sequences (Sample et al., 2019), each annotated with their ribosome loading, which is correlated with the expression level of the 5'UTR region.

TFBind-8 dataset contains sequences of length 8, consisting of four distinct nucleobases, each annotated with a ground-truth value of binding activity with human transcription factors. The dataset consists of $65,536$ sequences, i.e. all possible sequences of four nucleobases of length 8, i.e 4^8.

3.2. Tasks

The biological datasets described above can be formulated into sequence design tasks where the aim is to design sequences that maximise a desired property (which is typically expressed as the score of the sequences). Additionally, these tasks include an oracle model responsible for scoring *de novo* sequences generated by a design method.

For UTR and GFP datasets, the oracle employed is usually an ML model trained on the available dataset of sequences and their corresponding fitness values. In the following we describe the commonly used oracles for these tasks. In contrast, TF-Bind dataset provides values for every possible sequence, so there is no need for an oracle model, as *de novo* sequences are queried against the dataset.

GFP Several oracle models are commonly used for GFP, we consider the following three: 1) Design Bench Transformer (Trabucco et al., 2022) used in Trabucco et al. (2021); Jain et al. (2022); Kim et al. (2024), 2) TAPE (Rao et al., 2019)

[1] https://github.com/samsinai/FLEXS

used in Ren et al. (2022); Song & Li (2023); Wang et al. (2023), and 3) ESM-1b (Rives et al., 2021) with a fine-tuned head trained and used in Ren et al. (2022). We train the Trasnforer oracle ourselves (according to the Design Bench implementation), and use the available pre-trained checkpoints for TAPE and ESM-1b models. The Transformer oracle is trained on a random uniform $90/10\%$ train/validation split of the task dataset. TAPE and the ESM-1b are trained on task dataset sequences that are 3 mutational edits away from the wild-type sequence, and the remaining sequences of $4 - 15$ mutations constitute the validation set.

UTR There are two popular oracle architectures used for UTR: convolutional neural network (CNN) (Sample et al., 2019; Angermueller et al., 2020), and residual neural network (ResNet) (Trabucco et al., 2021; Kim et al., 2024). Following recent works, we use the ResNet oracle and employ the Design Bench parameters and training code (Trabucco et al., 2022). The ResNet oracle is trained on approximately 93% of the dataset ($260,000$ sequences) and is validated on the remaining 7%.

TFBind-8 This task has a ground-truth oracle as it includes a fully enumerated dataset that can be queried to retrieve the experimental scores of each sequence.

3.3. Sequence Generators

In this work, we evaluate 12 well-known design methods developed for biological sequence design: 1) Generative Flow Networks with Active Learning (GFN-AL[2], Jain et al. (2022)), 2) Bootstrapped training of score conditioned Generator (BootGen[3], Kim et al. (2024)) 3) Conditioning by Adaptive Sampling (CbAS, Brookes et al. (2019)), 4) Autofocused CbAS (Auto. CbAS, (Fannjiang & Listgarten, 2020)), 5) Bayesian optimization with a quasi-expected improvement acquisition function (BO-qEI, Wilson et al. (2018)), 6) Model Inversion Networks (MINs, Kumar & Levine (2020)), 7) Covariance Matrix Adaptation Evolution Strategy (CMA-ES, Hansen (2006)), 8) REINFORCE (Williams, 1992), 9)-11) Gradient Ascent (GA) with respect to a surrogate model, including two variations - taking the mean (GA Mean) and the minimum (GA Min) of the ensemble (Trabucco et al., 2022), 12) and Conservative Objective Models (COMs, Trabucco et al. (2021)). Implementations from Design Baselines repository[4].

After training, all methods are sampled to obtain a batch of 128 sequences. When performing seeded runs, each method is retrained and then sampled.

4. Evaluating *de novo* sequences with ML oracles

Our first contribution is to highlight the limitations in the evaluation process when leveraging ML-trained oracles for biological *de novo* sequence design.

As a practitioner, it is of utmost importance to have confidence in the oracles employed to provide ground truth values such that, ultimately, only the most promising *in silico de novo* sequences are proposed for (potentially expensive and time-consuming) evaluation *in vitro*. One question we ask here is whether the relative performance of different design methods vary, as we vary superfluous characteristics of the oracle. For example, are the leading design methods consistently performant with oracles trained across many random seeds or different architectures?

In Section 4.1, we reveal that the relative performance of 12 commonly used sequence design methods is highly sensitive to both (1) the random seed used to train the oracle and (2) the architecture of the oracle employed to score new sequences. These findings cast doubts on the conclusions of prior works, as it becomes challenging to determine whether a method is genuinely state-of-the-art or simply outperforms other methods due to inherent randomness in a specific oracle implementation. In Section 4.2, we dive into the reasons behind these inconsistencies, offering insights that suggest the poor generalization capabilities of commonly used ML oracle models may be a contributing factor.

4.1. What is state-of-the-art?

Many prior works that propose sequence design methods do so by training their own ML oracle to evaluate new sequences generated. They often leverage open-source implementations or implement their own architecturally different oracle which is trained on open-source datasets. As an example of the former, Design Bench open-source code for training oracles but not the weights of the oracle model itself, resulting in each study re-training their own oracle model. Since the Design Bench implementation is not seeded, each study ends up with different oracle model parameters. As an example of the latter, prior works use one of four architecturally different oracles (Design Bench Transformer (Trabucco et al., 2021; Kim et al., 2024), TAPE (Ren et al., 2022), and ESM-1b (Ren et al., 2022)) for the GFP task in prior works. Potential inconsistencies amongst the oracles may cause unreliable comparisons between methods.

Since many different oracles are used in prior works for any given sequence design task, minor variations in evaluated scores are to be expected. However, what we would expect, is low-variance across the highest scored sequences (across batches), and that under each oracle, there is a consistent ranking of relative performance of each design method. To

[2] https://github.com/MJ10/BioSeq-GFN-AL

[3] https://github.com/kaist-silab/bootgen

[4] https://github.com/brandontrabucco/design-baselines

Table 1. Relative ranking of 12 sequence design methods (descending order) across five random seed replications of the ML-oracle.

Seed 1	Seed 2	Seed 3	Seed 4	Seed 5
BootGen	BootGen	BootGen	BootGen	BootGen
CMA-ES	GA Min	GA Min	GA Min	GA Mean
GA Mean	BO-qEI	GA Mean	GA Mean	CMA-ES
GA Min	GA Mean	GA	GA	BO-qEI
COMs	GA	Auto. CbAS	BO-qEI	GA Min
Auto. CbAS	Auto. CbAS	CMA-ES	CMA-ES	Auto. CbAS
GA	COMs	BO-qEI	Auto. CbAS	COMs
BO-qEI	CMA-ES	COMs	COMs	GA
CbAS	MINs	MINs	MINs	CbAS
MINs	CbAS	CbAS	CbAS	MINs
REINFORCE	REINFORCE	REINFORCE	REINFORCE	REINFORCE
GFN-AL	GFN-AL	GFN-AL	GFN-AL	GFN-AL

Table 1. (a) UTR Design Bench oracle

Seed 1	Seed 2	Seed 3	Seed 4	Seed 5
BootGen	CbAS	MINs	MINs	MINs
REINFORCE	BootGen	BootGen	BootGen	BootGen
MINs	REINFORCE	CbAS	Auto. CbAS	Auto. CbAS
Auto. CbAS	GA Min	Auto. CbAS	REINFORCE	REINFORCE
CbAS	MINs	REINFORCE	CbAS	CbAS
COMs	GA Mean	GA Mean	GA Mean	GA Mean
GA Mean	Auto. CbAS	GA Min	GA Min	GA Min
GA Min	COMs	COMs	COMs	COMs
GA	GA	GA	GA	GA
GFN-AL	GFN-AL	GFN-AL	GFN-AL	GFN-AL
CMA-ES	CMA-ES	BO-qEI	CMA-ES	CMA-ES
BO-qEI	BO-qEI	CMA-ES	BO-qEI	BO-qEI

Table 1. (b) GFP Design Bench oracle

examine this, we test the consistency of the design methods evaluated against: (Experiment 1) a single oracle trained across five different random seeds, and (Experiment 2) three different ML oracle architectures (trained on the same task).

Experiment 1 To assess how robust the evaluation of design methods is under ML-based oracles, we select one oracle setup and vary the random seed used during training. If the oracle models were reliable and consistent, we would not expect this to affect the relative performance of the design methods. We demonstrate this on two sequence design tasks, UTR and GFP, and utilize the Design Bench ML oracles. Specifically, we re-train the oracles with five different random seeds, to assess how these random replications affect the performance of each design method. We train each of our 12 design methods on 8 random seeds as described in Section 3.3, and sample a batch of 128 sequences from each.

Results 1 In Table 1 we present the ranking of the maximum score achieved from sampled *de novo* batches for the 12 sequence design methods (averaged over 8 random seeds), where each column is a different seeded replication of the Design Bench oracle for (a) UTR and (b) GFP tasks. Table 3 (Appendix) shows the rankings of the 12 design methods for each of the 8 random seeds (columns) under a single oracle model instance. We collate the results in Figure 6 (Appendix) which shows the distribution of each design method's maximum sequence score under 5 seeded oracle models and 8 seeded designs.

For the GFP task, we see that three different design methods are considered SOTA in Table 1(b). Additionally, when considering the ranking of the 8 seeded design methods scored under a single oracle instance, we see from Table 3 that the results are highly variable, and there is a general lack of agreement on the relative performance across design seeds. Notably, for UTR, GFN-AL generated the highest scoring sequences under two seeds and the worst under the remaining 6 seeds. When aggregated across the 8 seeds in Table 1, we see some agreement among the different replications of the oracle models. Specifically, the methods that consistently generate low-scoring sequences are ranked in the bottom three, and for the UTR task, BootGen consistently generates SOTA *de novo* sequences. However, for the latter, we demonstrate in the following section that this is because BootGen is able to exploit a key limitation of ML-based oracles which is that they are unreliable in scoring sequences outside of their train set distribution. **This highlights our first contribution: evaluating design methods based on approximate, self-managed oracles, does not lead to insights into the design methods themselves, but rather the randomness inherent in the oracle evaluations.**

Experiment 2 To assess whether there is consistency among the relative performance of the 12 design methods when utilizing ML oracles, we compare oracles with different architectures (each optimised for the same task). We demonstrate this on the GFP sequence design task using 3 oracle models: (1) Design Bench Transformer, (2) TAPE, and (3) ESM-1b. Again, we generate a new batch of 128 sequences from each design method (under 8 random seeds) and evaluate their scores under each of the oracle models. We then rank the methods based on the maximum oracle score achieved within each generated batch, which is a common metric in the literature.

Results 2 Table 2 presents the rankings of the 12 methods for the three different oracle architectures. Although there is a general consensus that BootGen performs comparatively well and CMA-ES poorly, an overall lack of agreement among the rankings assigned by the different oracle models is evident. Noticeably, under ESM-1b GFN-AL and BO-qEL generate the highest scoring *de novo* sequences, however, under the Design Bench and TAPE oracles, rank these sequences as the tenth best, and BO-qEI as the lowest-scoring method. Similarly, MINs is the third best method under Design Bench, fifth best under TAPE, and eighth best under ESM-1b. **Our second takeaway is that allowing the community to evaluate generative sequence design methods by utilising different approximate ML oracles can be highly subjective to specific architectural choices.**

Table 2. The ranking of 12 design methods (descending order) for three different GFP oracles: Design Bench Transformer, TAPE, and fine-tuned ESM1b.

Design Bench	TAPE	ESM1b
BootGen	BootGen	GFN-AL
REINFORCE	REINFORCE	BO-qEI
MINs	CbAS	BootGen
Auto. CbAS	Auto. CbAS	REINFORCE
CbAS	MINs	CbAS
COMs	GA Min	Auto. CbAS
GA Mean	GA Mean	GA Min
GA Min	COMs	MINs
GA	GA	GA Mean
GFN-AL	GFN-AL	GA
CMA-ES	CMA-ES	COMs
BO-qEI	BO-qEI	CMA-ES

Concluding Remarks The inconsistency between design methods' relative performance across superfluous oracle-design choices highlights the potential pitfalls of relying on ML-based oracle for rigorous evaluation. Our results demonstrate that the choice of the oracle model heavily influences the perceived state-of-the-art performance across both DNA and protein sequence design tasks, emphasising the importance of either considering multiple oracles and/or including additional evaluation metrics for a more comprehensive and reliable assessment of *de novo* sequences.

4.2. Do ML oracles generalise?

We hypothesise that a major contributing factor to the variation of design methods' performance across different ML oracles, suggests inadequacies in the oracles themselves. Specifically, in their ability to generalize out-of-distribution (OOD) i.e. to new *de novo* sequences outside of the training data. In this section, we first evaluate the generalisation capabilities of three commonly used GFP sequence design oracles. Our results reveal poor generalisation performance across all oracle models. Consequently, we delve into the oracle training procedure to better understand the underlying reasons for the observed limitations.

Experiment 3 We analyse the performance of three commonly used ML oracles trained to score and evaluate *de novo* GFP protein sequence designs. We evaluate for both in-distribution and out-of-distribution error by utilizing the ground truth scores from the corresponding datasets. The three oracles are: 1) Design Bench Transformer, 2) TAPE, and 3) ESM-1b. Specifically, we investigate the error between the oracle's predicted score and the true experimental score taken directly from the dataset for both train and held-out validation data splits. Clearly, one expects low training set error, and understandably higher validation set error.

Results 3 Figure 1 presents the absolute error between each oracle scored sequence and the true dataset score for Design Bench Transformer, TAPE, and ESM-1b oracles. The Design Bench oracle demonstrates poor accuracy for both train and held-out validation sequences, suggesting that the oracle struggles to fit the training dataset accurately. Despite exhibiting low errors for some held-out sequences, the oracle overall does not generalise well to both *de novo* sequences, and sequences it has seen before.

The TAPE oracle achieves reasonable accuracy on the training dataset sequences; however, it shows poor generalisation to the held-out validation sequences. Considering the train/validation split (described in Section 3.2), TAPE fits the training dataset (sequences of up to 3 mutations from the GFP wildtype) more accurately, however, we see that it struggles to effectively generalize to the validation sequences (with greater than 3 mutations).

Finally, ESM-1b exhibits poor accuracy for sequences in both the train and validation sets, indicating a similar issue as observed with the Design Bench model: the model fails to fit the training dataset, and should not be relied on to provide robust ground truth scores for held-out sequences.

Analysis Overall, we find that all 3 commonly used oracles demonstrate poor generalisation capabilities on the GFP design task. It is interesting to note the similarity in the error distribution shown across all oracles. To better understand this, we highlight the data distribution of the available GFP ground truth scores in Appendix Figure 4 (centre).

The distribution is extremely unbalanced and bi-modal: one mode between 0 and 0.2 encompassing approximately 40% of the data; and the second mode between 0.6 to 0.95 representing roughly 60% of the data. In light of this unbalanced data distribution, and the common practice of taking random data splits, one hypothesis is that the ML oracles converge to simply predicting the score of every sequence to one of the two modes. Consequently, the error plots reveal two distinct peaks, each reflecting increasing error as the sequences vary from these two modes.

4.3. Analysing generalisation via state space coverage

Biological sequence design tasks typically have a combinatorially large search space that grows exponentially with the sequence length. By contrast, the datasets available for many of these tasks cover only a tiny fraction of this space.

We investigate this phenomenon using the UTR task, with a sequence length of 50, and therefore state-space of 4^{50} possible combinations. The available UTR labelled dataset is 280,000 sequences representing less than 0.001% of the total possible space. In the previous section, we demonstrated the oracle's lack of ability to generalise OOD. A resulting outcome of this, is that for example, the Design Bench ML oracle can predict surprisingly high scores (outside of the dataset range) of 0.78 and 0.86 to sequences composed entirely of either adenine (A) or thymine (T) bases. The highest score in the available dataset is 0.73.

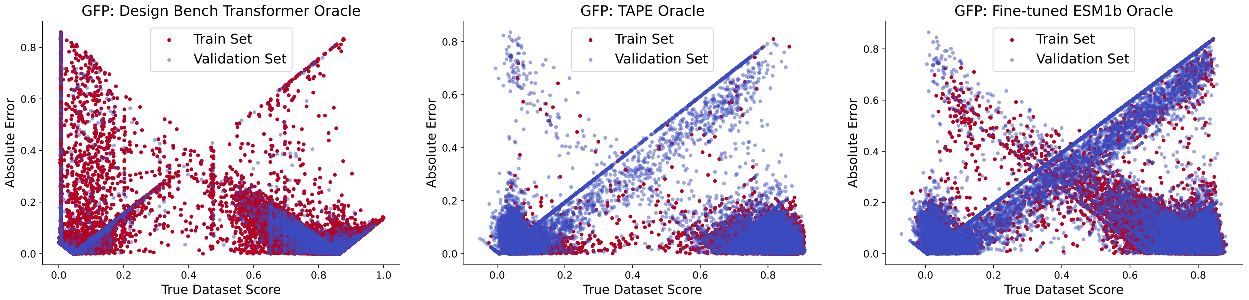

Figure 1. Error between the oracle predictions and true dataset scores per sequence for the train (red) and held-out validation (blue) datasets for three GFP oracles: (left) Design-Bench transformer, (middle) TAPE, (right) ESM-1b fine-tuned on GFP dataset.

Due to the unavailability of ground truth scores for all sequences in the UTR state space, we cannot fully assess the oracle generalisation to out-of-distribution sequences without further restricting the training dataset. However, we can recreate an equivalent ML oracle on a smaller DNA sequence design task, for example, leveraging the TFBind-8 DNA dataset that spans the entire state space 4^8 (= 65 536) sequences.

Experiment 4 To faithfully recreate the UTR Design Bench ML oracle using the TFBind-8 dataset, we train an equivalent ResNet oracle on 1% of the TFBind-8 dataset, with an equivalent random data split strategy. (Note 1% train split is an overestimation as compared to the UTR task data splits).

Results 4 Figure 2 illustrates the absolute error between the scores in the dataset and the oracle predicted scores, for both the training dataset (1% state space coverage) and held-out validation set (remaining 99% state space coverage). Given that TFBind-8 provides experimentally computed ground truth scores directly from a wet lab experiment, (albeit subject to inherent noise), comparing these scores with those predicted by the ML oracle offers insight into how reliably the Design Bench-inspired ML oracle can possibly represent the wet lab from such small state-space coverage. Our results highlight a larger error for held-out validation sequences, indicating poor generalisation of the oracle to unseen data. Notably, the oracle exhibits significant errors at the extremes of the score distribution, corresponding to sequences that are either truly high- or low-performing – which from a design task perspective are the sequences a practitioner would be most interested in robustly scoring *in silico*.

Concluding Remarks We have demonstrated that within common DNA and protein sequence design tasks, training an ML-based oracle on a tiny fraction, e.g. less than 0.001%, of the possible space of sequences, has a tendency to cause poor accuracy in scoring sequences beyond its training distribution (and often within!). This issue is particularly highlighted with the TAPE oracle for GFP, which fails to generalize even to sequences with 3-15 mutations, let alone the remaining 230 mutations required to evaluate on entirely new *de novo* generated protein sequences. Previous studies often make the assumption that the ML-based oracles accurately represent wet lab scores, and thus, use it to evaluate the sequences generated by their method. However, since some methods do not constrain their generation process to align with the dataset distribution, oracle are forced to score wildly OOD sequences, leading to unreliable evaluations.

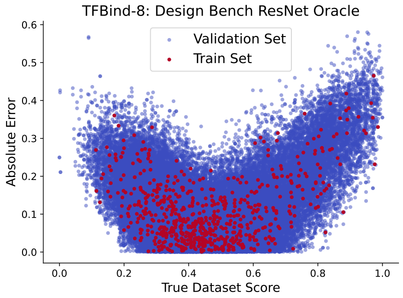

Figure 2. TFBind-8: Absolute error between the Design Bench-inspired ML oracle predictions and ground truth dataset scores for train (red) and held-out validation (blue) datasplits.

5. Leveraging Biophysical Measures for Improved Sequence Design

In the previous section, we demonstrated that popular ML oracles struggle to generalise OOD and can assign high scores to seemingly implausible sequences. In this section, we propose a strategy based on the hypothesis that a single ML oracle is not reliable enough to be used in isolation when evaluating a design method due to its aforementioned limitations. That is, we introduce a suite of biophysical measures that can assist practitioners in reducing the sequence state space and alleviate the oracle being evaluated OOD on *de novo* generated sequences.

When practitioners generate a batch of sequences to be sent for evaluation in a wet lab, it is crucial to ensure those sequences are good candidates and biologically sound, to avoid wasting resources. To achieve this, we propose that in practical settings, a suite of biophysical measures can be employed that leverage the data-distributions in the available data. By applying these measures to a batch of new sequences, we can automatically identify those with a higher likelihood of being biologically valid and more likely to succeed in wet lab experiments. These measures essentially reduce the search space of possible sequences by grounding the generation to regions associated with available data.

5.1. Suite of Biophysical Measures

The suite of measures represents a set of checks to validate whether sequences are biologically plausible (with a high degree of confidence). Whilst this is not an exhaustive set of biophysical measures one could use, we aimed to include somewhat general measures indicative of biological fitness, for both DNA and protein sequences.

DNA Measures Three DNA measures are *coverage*, *guanine-cytosine (GC) content*, and *homopolymer*. *Coverage* refers to the proportion of each of the four nucleobases represented in the sequence. *GC content* measures the percentage of guanine and cytosine bases in a sequence. GC content of a sequence can significantly impact thermostability which is a vital aspect of success in a wet lab. A *homopolymer* is a stretch of consecutive identical nucleobases in a sequence. Long homopolymers can cause issues in downstream applications, such as PCR and sequencing, and thus, are good indicators of artificial sequences.

Protein Measures 5 protein measures are *molecular weight*, *aromaticity*, *isoelectric point*, *grand average of hydropathy (gravy)*, and *instability index*. *Molecular weight* is the sum of the atomic weights of all atoms in a protein molecule and can influence the protein's stability, folding, and function. *Aromaticity* refers to the presence and distribution of aromatic amino acids (tryptophan, tyrosine, and phenylalanine) in a sequence. Aromatic residues play critical roles in protein stability, folding, and interactions with other molecules. The *isoelectric point* is the pH at which a protein has a net charge of zero (i.e., an equal number of positively and negatively charged residues) and this is a crucial factor in protein solubility. *Gravy* is a measure of the overall hydrophobicity or hydropathy of a protein sequence and provides insights into the protein's structural and functional properties. The *instability index* is a measure of a protein's susceptibility to degradation or denaturation. A high (low) instability index indicates that the protein is likely to be unstable (stable) and have a shorter (longer) half-life.

5.2. Evaluating with Biophysical Measures

Evaluation Procedure For each biological measure, we introduce an acceptable range under a reference dataset as the 99% middle quantile, and consider a sequence valid if it lies within that range for all measures. During our work, we considered two alternatives for the reference dataset: (1) the specific task dataset, and (2) a general distribution of natural sequences. Both strategies provide a data distribution from valid and biologically plausible sequences taken from the available data that we can leverage to reduce *de novo* generation of implausible sequences.

To incorporate these measures into the evaluation of our sequence design methods, we sample a batch of 128 sequences and evaluate each sequence with respect to the suite of measures to determine whether it is likely to be a plausible sequence. We report the percentage of valid sequences for each design method, where clearly, higher is better.

Experiment 5 We evaluate 12 sequence design methods introduced in Section 3.3 on both DNA (UTR) and protein (GFP) sequence design tasks. Specifically, for UTR, we leverage two reference datasets: (1) the entire UTR task dataset (280,000 sequences), and (2) the more general GENCODE database Harrow et al. (2012).[5] The protein sequences designed for GFP are evaluated using the five protein measures against a reference dataset of the entire GFP task dataset (51,715).[6] Since the task dataset already contains valid GFP sequences, we can directly compare the generated sequences to those in the dataset, ensuring meaningful property ranges.

Results 5 In Figure 3 we classify the valid (solid) and invalid (shaded) generated UTR and GFP sequences respectively for each of the 12 design methods by applying our proposed suite of biological measures to the 128 *de novo* generated batch. The percentage of valid sequences per method is displayed at the top of each bar.

For UTR sequences, the left plot presents valid sequences with respect to the UTR task dataset, while the centre plot is with respect to the more general GENCODE human UTR dataset. For the task dataset, the methods that generate a higher proportion of biologically meaningful sequences are Auto. CbAS, CbAS, BO-qEI, REINFORCE, and MINs (in

[5] GENCODE identifies and classifies gene features in human and mouse genomes. Since the UTR task is designed for humans, we create a reference dataset by compiling all human chromosomes annotated with UTRs that are of lengths between 40 and 60 (to ensure comparability with the task-specified 50-length UTRs). This results in a final dataset of 34,060 sequences.

[6] A more generic reference protein sequence dataset is not appropriate for this setting, since valid GFP sequences must possess specific properties. For instance, GFP has an unusual covalent bond in its chromophoric Tyr residue, making its functional site's physicochemical microenvironment likely to be quite different from other proteins. In fact, classical atomic simulations designed for general proteins need adjustments to accurately estimate the energetics of GFP (Breyfogle et al., 2023). Consequently, measures and metrics derived from or applicable to general proteins might not accurately represent the unique chemical environment of GFP.

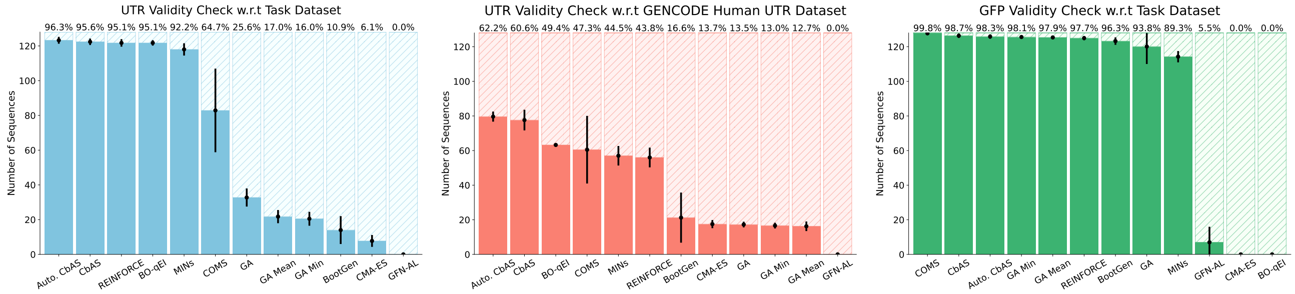

Figure 3. Generated sequences classified by DNA (left and centre) and protein (right) suite of biological measures, for 12 sequence design methods with respect to the task dataset (left and right) and the GENCODE database (centre).

descending order), with over 90% of sequences in their generated batch being valid. COMs obtains a batch with 64.7% valid sequences, and the remaining methods, GA, GA Mean, GA Min, BootGen, CMA-ES, and GFN-AL (in descending performance order), have less than 30% valid sequences, with strikingly GFN-AL generating 0% valid sequences. This highlights that these methods generate sequences with biological properties significantly different from the task reference dataset. Consequently, we can assume that these methods generate out-of-distribution (OOD) sequences compared to the task dataset distribution. Given the oracle's poor generalization to OOD sequences, it is worth considering whether invalid sequences should be included in the final batch when reporting a method's performance. With respect to the GENCODE reference dataset, the top-performing methods are Auto. CbAS, CbAS, BO-qEI, COMs, MINs, and REINFORCE (in descending order), similar to the task dataset results. However, these six methods exhibit much lower performance when evaluated against the GENCODE dataset compared to the task dataset. It is also interesting to note that BootGen and CMA-ES are more competitive than GA and its variations on this dataset, however, GFN-AL performs consistently poor, achieving 0% valid sequences.

For the generated protein sequences, 9 out of 12 methods achieve over 85% valid GFP sequences in the batch, including COMs, CbAS, Auto. CbAS, GA Min, GA Mean, REINFORCE, BootGen, GA, and MINs, in descending order. Additionally, GFN-AL exhibits poor performance at 5.5% valid sequences, and the remaining methods, Bo-qEI, CMA-ES, GA and its variations, have no valid sequences; this is not surprising as all three methods obtain very low performance when scored by the oracle model.

An alternative approach to determining the biophysical validity of *de novo* sequences is to compute the distributional conformity score (DCS) of the sequences to a reference dataset (Frey et al., 2024). Figure 7 (Appendix) shows the percentage of valid sequences per method for each task and reference dataset computed using the DCS of each sequence. Comparing these results with Figure 3, we observe a similar trend between the methods that consistently generate a high percentage of valid sequences, supporting our evaluation approach and results.

6. Conclusion

Our work examines the reliability and consistency of the *in silico* biological sequence design benchmarks. Sequence design methods are commonly evaluated using an ML oracle model, trained a limited dataset of sequences, to score *de novo* generated sequences.

Our first contribution demonstrates that oracles with different seeded runs and architectures result in conflicting rankings of 12 sequence design methods. This lack of consensus among oracles raises concerns regarding the reliability of the oracle models, and our analysis suggests their poor generalisation to out-of-distribution sequences as a key limitation. Our second contribution introduces a set of biophysical measures to supplement the evaluation procedure. These metrics assess the biological feasibility of *de novo* sequences and effectively limit the space of out-of-distribution sequences the oracle needs to score, thereby improving the robustness of the design procedure.

In summary, our work highlights the potential limitations in the current evaluation procedure and presents biologically grounded measures to improve the robustness of design benchmarks, with the ultimate goal of enhancing *in silico* design methods. The most significant and challenging direction for future work lies in improving the oracles. With the emergence of more accurate nucleotide (Dalla-Torre et al., 2023) and protein language models (Lin et al., 2022), they should be considered for fine-tuning and application as task-specific oracles. Additionally, the introduced biophysical measures are generic and thus, applicable to all DNA or protein tasks. While these help filter out implausible sequences, developing more accurate and task-specific measures can further refine the state space and increase the likelihood of generating successful, biologically fit sequences. We leave this exploration for future work.

References

Angermueller, C., Dohan, D., Belanger, D., Deshpande, R., Murphy, K., and Colwell, L. Model-based reinforcement learning for biological sequence design. In *International conference on learning representations*, 2019.

Angermueller, C., Belanger, D., Gane, A., Mariet, Z., Dohan, D., Murphy, K., Colwell, L., and Sculley, D. Population-based black-box optimization for biological sequence design. In *International conference on machine learning*, pp. 324–334. PMLR, 2020.

Breyfogle, K. L., Blood, D. L., Rosnik, A. M., and Krueger, B. P. Molecular dynamics force field parameters for the egfp chromophore and some of its analogues. *The Journal of Physical Chemistry B*, 127(26):5772–5788, 2023.

Brookes, D., Park, H., and Listgarten, J. Conditioning by adaptive sampling for robust design. In *International conference on machine learning*, pp. 773–782. PMLR, 2019.

Buttenschoen, M., Morris, G. M., and Deane, C. M. Posebusters: Ai-based docking methods fail to generate physically valid poses or generalise to novel sequences. *Chemical Science*, 2024.

Chen, C., Zhang, Y., Liu, X., and Coates, M. Bidirectional learning for offline model-based biological sequence design. In *International Conference on Machine Learning*, pp. 5351–5366. PMLR, 2023.

Dalla-Torre, H., Gonzalez, L., Mendoza-Revilla, J., Carranza, N. L., Grzywaczewski, A. H., Oteri, F., Dallago, C., Trop, E., Sirelkhatim, H., Richard, G., et al. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, pp. 2023–01, 2023.

Fannjiang, C. and Listgarten, J. Autofocused oracles for model-based design. *Advances in Neural Information Processing Systems*, 33:12945–12956, 2020.

Frey, N. C., Berenberg, D., Zadorozhny, K., Kleinhenz, J., Lafrance-Vanasse, J., Hotzel, I., Wu, Y., Ra, S., Bonneau, R., Cho, K., et al. Protein discovery with discrete walk-jump sampling. *International Conference on Learning Representations*, 2024.

Hansen, N. The cma evolution strategy: a comparing review. *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms*, pp. 75–102, 2006.

Harris, C., Didi, K., Jamasb, A. R., Joshi, C. K., Mathis, S. V., Lio, P., and Blundell, T. Benchmarking generated poses: How rational is structure-based drug design with generative models? *arXiv preprint arXiv:2308.07413*, 2023.

Harrow, J., Frankish, A., Gonzalez, J. M., Tapanari, E., Diekhans, M., Kokocinski, F., Aken, B. L., Barrell, D., Zadissa, A., Searle, S., et al. Gencode: the reference human genome annotation for the encode project. *Genome research*, 22(9):1760–1774, 2012.

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 *International Conference on Machine Learning*, pp. 9786–9801. PMLR, 2022.

Kim, M., Berto, F., Ahn, S., and Park, J. Bootstrapped training of score-conditioned generator for offline design of biological sequences. *Advances in Neural Information Processing Systems*, 36, 2024.

Kumar, A. and Levine, S. Model inversion networks for model-based optimization. *Advances in neural information processing systems*, 33:5126–5137, 2020.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., dos Santos Costa, A., Fazel-Zarandi, M., Sercu, T., Candido, S., et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022.

Notin, P., Kollasch, A., Ritter, D., Van Niekerk, L., Paul, S., Spinner, H., Rollins, N., Shaw, A., Orenbuch, R., Weitzman, R., et al. Proteingym: large-scale benchmarks for protein fitness prediction and design. *Advances in Neural Information Processing Systems*, 36, 2024.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, P., Canny, J., Abbeel, P., and Song, Y. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019.

Ren, Z., Li, J., Ding, F., Zhou, Y., Ma, J., and Peng, J. Proximal exploration for model-guided protein sequence design. In *International Conference on Machine Learning*, pp. 18520–18536. PMLR, 2022.

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021.

Sample, P. J., Wang, B., Reid, D. W., Presnyak, V., McFadyen, I. J., Morris, D. R., and Seelig, G. Human 5' utr design and variant effect prediction from a massively parallel translation assay. *Nature biotechnology*, 37(7): 803–809, 2019.

Sarkisyan, K. S., Bolotin, D. A., Meer, M. V., Usmanova, D. R., Mishin, A. S., Sharonov, G. V., Ivankov, D. N., Bozhanova, N. G., Baranov, M. S., Soylemez, O., et al. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401, 2016.

Song, Z. and Li, L. Importance weighted expectation-maximization for protein sequence design. In *International Conference on Machine Learning*, pp. 32349–32364. PMLR, 2023.

Spinner, H., Kollasch, A. W., and Marks, D. S. How well do generative protein models generate? In *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2024.

Tagasovska, N., Park, J. W., Kirchmeyer, M., Frey, N. C., Watkins, A. M., Ismail, A. A., Jamasb, A. R., Lee, E., Bryson, T., Ra, S., et al. Antibody domainbed: Out-of-distribution generalization in therapeutic protein design. *International Conference on Learning Representations*, 2024.

Trabucco, B., Kumar, A., Geng, X., and Levine, S. Conservative objective models for effective offline model-based optimization. In *International Conference on Machine Learning*, pp. 10358–10368. PMLR, 2021.

Trabucco, B., Geng, X., Kumar, A., and Levine, S. Design-bench: Benchmarks for data-driven offline model-based optimization. In *International Conference on Machine Learning*, pp. 21658–21676. PMLR, 2022.

Wang, Y., Tang, H., Huang, L., Pan, L., Yang, L., Yang, H., Mu, F., and Yang, M. Self-play reinforcement learning guides protein engineering. *Nature Machine Intelligence*, 5(8):845–860, 2023.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Wilson, J., Hutter, F., and Deisenroth, M. Maximizing acquisition functions for bayesian optimization. *Advances in neural information processing systems*, 31, 2018.

A. Score Distributions of Sequence Datasets

Figure 4 illustrates the distribution of the scores corresponding to each sequence in the following datasets: UTR (left), GFP (centre), and TFBind-8 (right).

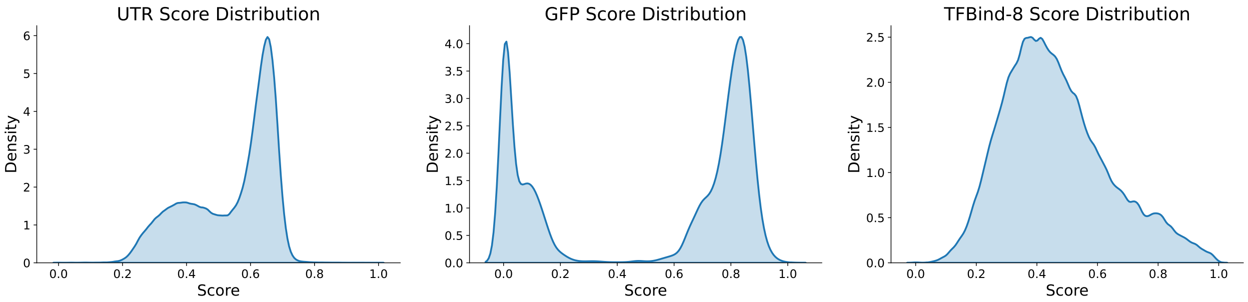

Figure 4. The distribution of scores for the UTR (left), GFP (centre), and TFBind-8 (right) datasets.

B. Additional Sequence Design Results under the Design Bench Oracle

To assess the robustness of *in silico* design methods, we trained 12 design methods (described in Section 3.3) using 8 different seeds for both the UTR and GFP tasks, and evaluate them using the Design Bench oracle model. The results, presented in Table 3, show the rankings of the 12 methods for (a) UTR and (b) GFP tasks with 8 seeded replications. The inconsistency in the rankings suggests that these methods are sensitive to the choice of training seed.

Table 3. Relative ranking of 12 sequence design methods (descending order) across 8 random seed replications of the methods, evaluated under the Design Bench oracle.

Table 3. (a) UTR

Seed 1	**Seed 2**	**Seed 3**	**Seed 4**	**Seed 5**	**Seed 6**	**Seed 7**	**Seed 8**
GFN-AL	BootGen	GFN-AL	BootGen	BootGen	BootGen	BootGen	BootGen
BootGen	CMA-ES	BootGen	GA Min	GA Mean	CMA-ES	GA Min	COMs
CMA-ES	BO-QEI	CMA-ES	BO-QEI	GA Min	GA Mean	CMA-ES	GA Min
COMs	Auto. CbAS	Auto. CbAS	CbAS	BO-QEI	COMs	GA Mean	GA Mean
GA Mean	COMs	GA Mean	Auto. CbAS	CMA-ES	BO-QEI	REINFORCE	CMA-ES
GA	GA Min	CbAS	MINs	MINs	GA	MINs	GA
MINs	GA	GA	COMs	GA	GA Min	CbAS	REINFORCE
GA Min	CbAS	BO-QEI	GA Mean	Auto. CbAS	Auto. CbAS	Auto. CbAS	Auto. CbAS
CbAS	MINs	REINFORCE	GA	CbAS	CbAS	GA	MINs
Auto. CbAS	GA Mean	GA Min	CMA-ES	COMs	REINFORCE	COMs	CbAS
BO-QEI	REINFORCE	MINs	REINFORCE	REINFORCE	MINs	BO-QEI	BO-QEI
REINFORCE	GFN-AL	COMs	GFN-AL	GFN-AL	GFN-AL	GFN-AL	GFN-AL

Table 3. (b) GFP

Seed 1	**Seed 2**	**Seed 3**	**Seed 4**	**Seed 5**	**Seed 6**	**Seed 7**	**Seed 8**
CbAS	REINFORCE	MINs	BootGen	REINFORCE	Auto. CbAS	Auto. CbAS	MINs
Auto. CbAS	BootGen	BootGen	CbAS	BootGen	BootGen	REINFORCE	REINFORCE
REINFORCE	CbAS	REINFORCE	REINFORCE	CbAS	MINs	MINs	CbAS
BootGen	Auto. CbAS	COMs	MINs	Auto. CbAS	REINFORCE	COMs	BootGen
MINs	MINs	CbAS	COMs	MINs	COMs	CbAS	Auto. CbAS
COMs	COMs	Auto. CbAS	Auto. CbAS	COMs	CbAS	BootGen	COMs
GFN-AL	GFN-AL	GFN-AL	CMA-ES	CMA-ES	GFN-AL	GFN-AL	GFN-AL
CMA-ES	CMA-ES	CMA-ES	GA Min	GA Mean	CMA-ES	CMA-ES	CMA-ES
GA Mean	GA Min	GA	GA Mean	GA Min	GA Mean	GA	GA Mean
GA Min	GA Mean	GA Mean	BO-qEI	GA	BO-qEI	GA Min	GA Min
BO-qEI	BO-qEI	GA Min	GFN-AL	GFN-AL	GA Min	BO-qEI	BO-qEI
GA	GA	BO-qEI	GA	BO-qEI	GA	GA Mean	GA

Additionally, to understand how random seeded replications of both the design methods and the oracles impact the maximum score in the final batch of *de novo* generated sequences, we plot the distribution of the maximum score under these replications. Specifically, Figure 5 shows the distribution of the maximum score under 8 replications of the design methods and three different GFP oracles: DesignBench (Trabucco et al., 2022), ESM-1b (Rives et al., 2021), and TAPE (Ren et al., 2022). Figure 6 depicts the distribution of the maximum score under 8 replications of the design methods and 5 replications of the DesignBench ML oracle for both UTR and GFP.

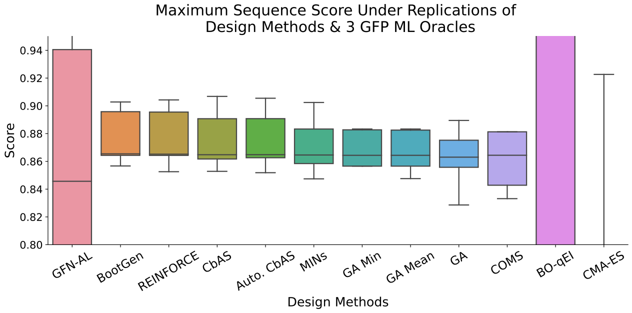

Figure 5. Distribution of the maximum score in the batch of *de novo* sequences generated under 8 replications of the design methods and 3 different GFP oracles.

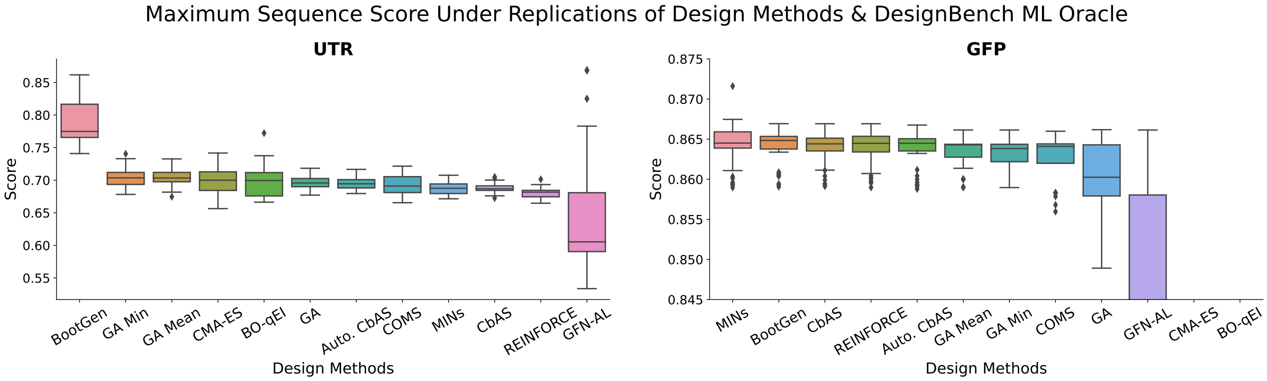

Figure 6. Distribution of the maximum score in the batch of *de novo* sequences generated under 8 replications of the design methods and 5 replications of the DesignBench ML oracles.

C. Additional Results under the Biophysical Measures

Frey et al. (2024) introduced the distributional conformity score (DCS) to improve the quality of *de novo* generated sequences with the aim that the score directly relates to the probability of generating real, biophysically valid proteins. To verify our approach of denoting *de novo* generated sequences as valid, we recompute the validity of the sequences generated by each of the 12 design methods under the DCS. The results, illustrated in Figure 7, directly match our results in Figure 3 for all tasks and reference datasets.

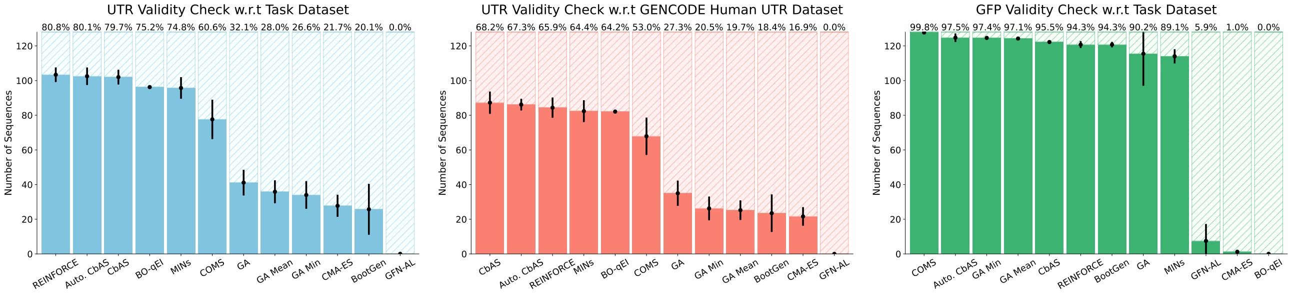

Figure 7. Generated sequences classified by DNA (left and centre) and protein (right) suite of biological measures, for 12 sequence design methods with respect to the task dataset (left and right) and the GENCODE database (centre).