ACS SyntheticBiology

pubs.acs.org/synthbio Research Article

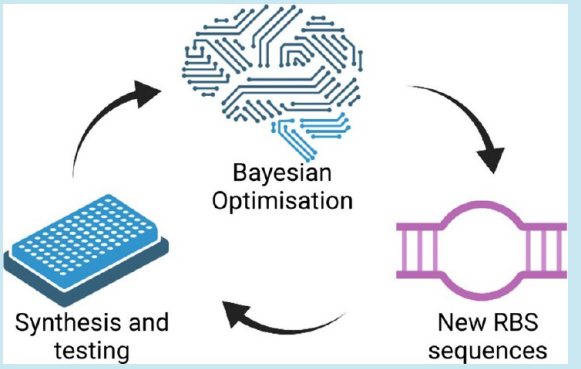
Machine Learning Guided Batched Design of a Bacterial Ribosome Binding Site

Mengyan Zhang, Maciej Bartosz Holowko, Huw Hayman Zumpe, and Cheng Soon Ong*

Cite This: *ACS Synth. Biol.* 2022, 11, 2314–2326 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: Optimization of gene expression levels is an essential part of the organism design process. Fine control of this process can be achieved by engineering transcription and translation control elements, including the ribosome binding site (RBS). Unfortunately, the design of specific genetic parts remains challenging because of the lack of reliable design methods. To address this problem, we have created a machine learning guided Design–Build–Test–Learn (DBTL) cycle for the experimental design of bacterial RBSs to demonstrate how small genetic parts can be reliably designed using relatively small, high-quality data sets. We used Gaussian Process Regression for the Learn phase of the cycle and the Upper Confidence Bound multiarmed bandit algorithm for the Design of genetic variants to be tested in vivo. We have integrated these machine learning algorithms with laboratory automation and high-throughput processes for reliable data generation. Notably, by Testing a total of 450 RBS variants in four DBTL cycles, we have experimentally validated RBSs with high translation initiation rates equaling or exceeding our benchmark RBS by up to 34%. Overall, our results show that machine learning is a powerful tool for designing RBSs, and they pave the way toward more complicated genetic devices.



KEYWORDS: *machine learning, optimization, genetic part design, ribosome binding site*

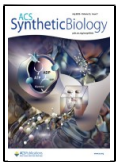
1. INTRODUCTION

One of the main tenets of synthetic biology is the design, evaluation, and standardization of genetic parts.[1−3] A central challenge is part design, which is understood as modifying the sequence of a genetic part for it to meet specific requirements. Genetic parts are the units which are ultimately combined into more complex genetic circuits that produce desired functions in the target organisms. This is usually done in terms of the Design–Build–Test–Learn (DBTL) cycle, where a given genetic part or organism is continually improved through an iterative process. This cycle involves designing (D) new DNA sequences to achieve a desired property in computer-aided design software, then physically building (B) new constructed variants and testing (T) using an analytical instrument in a laboratory. Computer modeling can be used to learn (L) and predict the characteristics of a genetic part.[4,5] Most of these computer models are based on either the thermodynamic properties of the involved molecules (DNA, RNA, proteins, etc.) or empirically obtained values describing a relevant design property, like translation initiation rate (TIR) in the case of ribosome binding sites (RBSs).[6−8] However, de novo design of small genetic elements is challenging because of unknown relationships between their sequence and performance.

In this paper, we propose a machine learning guided Design–Build–Test–Learn cycle for the experimental design of bacterial RBSs. This consists of two distinct types of machine learning methods, one in the Learn phase and a second in the Design phase. We show how small genetic parts can be reliably designed using even relatively small, but high-quality data sets. This work focuses on RBS part design and TIR prediction, rather than looking at its wider genetic context and impact on the general performance of the cell. As the RBS is one of the key genetic elements controlling protein expression in bacteria, it is a suitable target for establishing a workflow that could be potentially translated to more complicated systems.

In the Design phase of the DBTL cycle, designers often fine-tune the characteristics of parts to give the resulting strains their desired properties. The ability to predict a characteristic of a genetic part (for example using a machine learning method) only provides inputs to the problem. The designer still needs to choose from the large number of possible variants to Build. For instance, to increase the yield of a protein,

Received: January 11, 2022
Published: June 15, 2022



https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314–2326

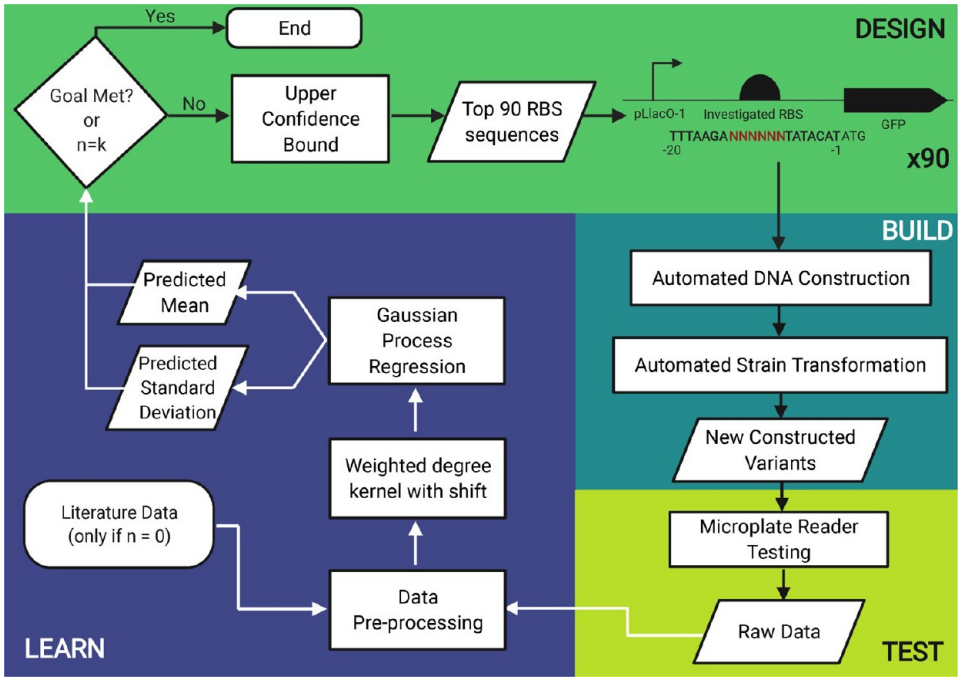


Figure 1. Flowchart of the machine learning-based experimental design. The RBS design is recommended by the Upper Confidence Bound multiarmed Bandit algorithm. After generating the recommendations, the RBSs are built and tested using automated laboratory methods allowing for rapid construction and testing at scale. Finally, the obtained results are fed back to the Gaussian Process Regression prediction algorithm in the Learn phase. *n* is the current design round and *k* is the maximum number of rounds allowed by time and/or money. In regards to the "Goal Met?" condition, the goal in our case was to find sequences with TIR significantly higher than benchmark, but the goal can be generalized to fit the requirements of the user.

increasing the TIR of the RBS responsible for translation of that protein could be targeted. Hence, the goal of the Design phase of TIR is to select a small set of RBS sequences from the design space (i.e., all possible DNA sequences) based on the predictions from Learn phase. However, since the computational predictions are not perfectly accurate, there is uncertainty about which potential RBS sequence has the desired property (for example highest protein yield). The designer could use the mean predicted TIR values and choose the best ones to exploit the knowledge modeled by the computational predictions. But the designer may also want to explore regions of DNA design space where the predictions are highly uncertain (e.g., as measured by the predicted TIR standard deviation) and could yield high payoff. Therefore, a major challenge of the Design phase is to balance the exploitation and exploration, which is addressed by one class of machine learning approaches called multiarmed bandits.[9] As we will see in this paper, multiarmed bandits are well suited to solving the challenge of recommending a small set of RBS sequences in the Design phase of the DBTL cycle.

One way to generate predicted TIR values is to use existing RBS calculators.[8] Three main RBS calculators were surveyed, all of which predict the TIR based on the thermodynamic properties of the RBS and the ribosome.[10−12] Predictions reported from all of these models range from relatively good ($R^2 > 0.8$) to low ($R^2 < 0.2$) depending on the data set.[13] This may be caused by the following: (i) they rely on calculations of free energies, which can be difficult to estimate with high precision, (ii) in general, one of the best ways to improve the models' accuracy is by increasing the number of phenomena taken into account, which can lead paradoxically to decreased model accuracy due to accumulation of errors,[14] and (iii) by using deterministic coefficients to calculate energies, one disregards the often stochastic nature of processes in cells, which can potentially increase prediction error.[15] There is also evidence that binding energy calculations may be poor predictors of RBS strength.[16,17] This is reinforced by studies suggesting that RNA secondary structure is potentially a more important feature in TIR determination.[14,18] This suggests that multiple interactions determine the mRNA-ribosome binding, and predicting TIR from the genomic sequence is still challenging.

Recent work has explored the use of machine learning predictors to learn from historical data and generate predictions for use in synthetic biology, vastly improving the DBTL cycle's performance.[19−21] This work leverages the exponential increase in experimental data produced in synthetic biology[22] to improve predictors in the Learn phase of the DBTL cycle. For example, Jervis, Carbonell et al.[23] used support vector machines and neural networks to optimize production of monoterpenoids in *Escherichia coli*. Similarly,[24] others have used a number of machine learning approaches to analyze time-series multiomics data to predict metabolic pathway behavior. Deep learning techniques have also been successfully used to analyze large synthetic biology data sets.[25−27] Recall that the multiarmed bandit approach balances the exploration−exploitation trade off by using predictive uncertainty. The machine learning predictors mentioned, as well as existing RBS calculators, do not yet provide uncertainty levels to guide exploration with the notable exception of ART.[20] Furthermore, large data sets may not be available for a genetic part of interest, for instance the RBS. Hence there is a need to develop methods for training predictors on smaller data sets and generate predictions for both their mean values

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

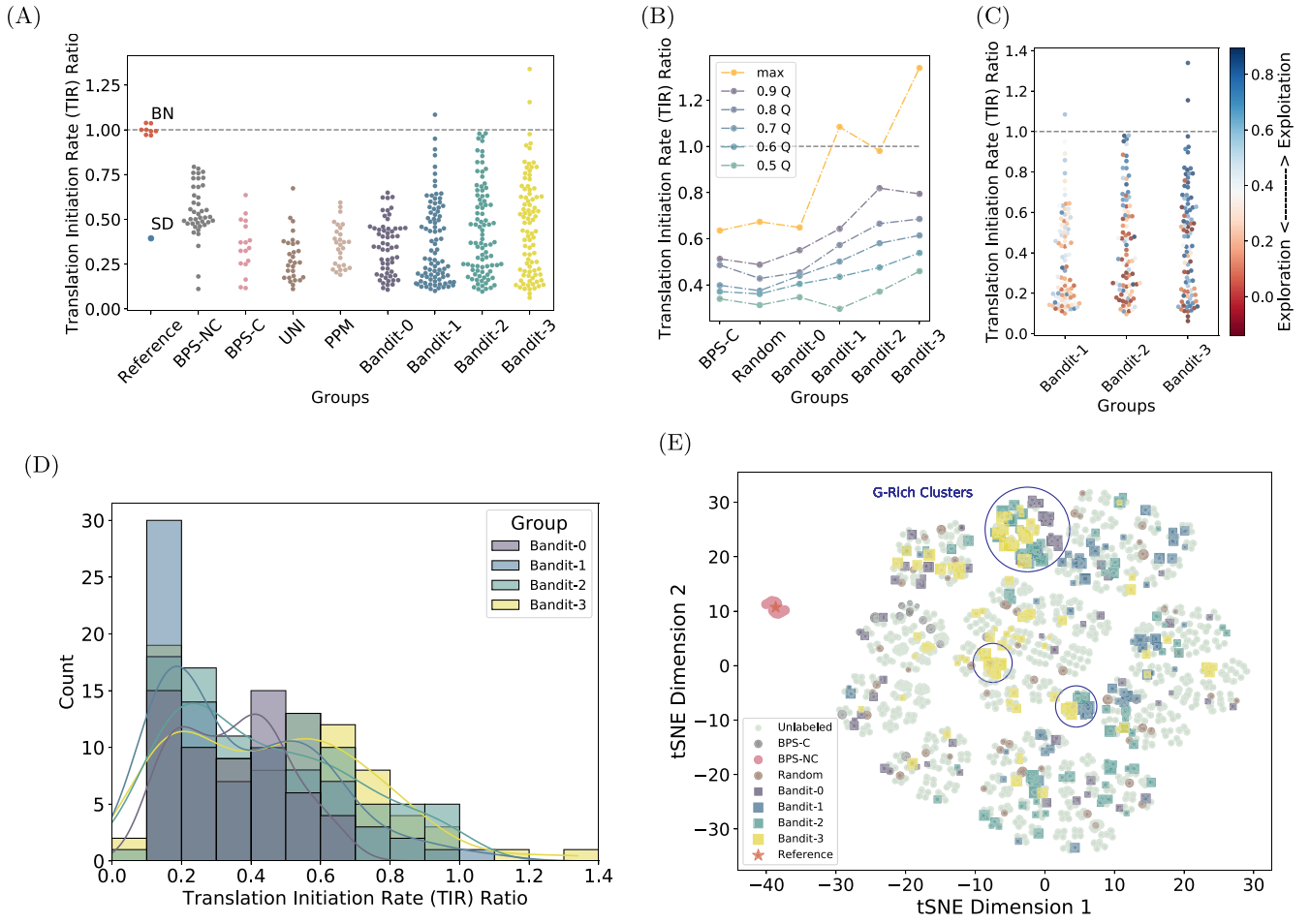


Figure 2. TIRs of RBS groups examined in this study. (A) Swarm plot showing the obtained TIRs divided into RBS groups. BPS-NC: Base-by-base changes in the noncore region. BPS-C: Base-by-base changes in the core region. UNI: Randomly generated sequences with uniform distribution. PPM: Randomly generated sequences with distribution following the Position Probability Matrix for all natural RBS in *E. coli*. Bandit-0/1/2/3: Bandit algorithm generated results for Round 0, 1, 2, and 3 respectively. SD: Shine−Dalgarno sequence. Dashed line is set to 1 and represents the averaged benchmark sequence TIR for that group. BN: Benchmark sequences for all plates (not all are exactly 1 because of them being shown as separate samples rather than per round averages). (B) Line plot showing TIR obtained in a given quantile (Q) of results divided into groups as in (A). UNI and PPM are merged into Random group, and BPS-NC is not shown because of changes being made outside the core in that group. (C) Exploitation vs Exploration for Bandit 1−3. Blue-hued points represent exploitation, those hued red represent exploration. (D) Histogram with kernel density estimations (KDE) showing distributions of TIRs for Bandit groups. (E) t-SNE plot showing the relative distances between sequences in our design space as calculated by our kernel function (weighted degree kernel with shift). The Unlabeled points represent the RBS sequences in the design space that have not been tested. The area of the marker corresponds to the experimentally obtained TIR value. The TIR results in all subplots are shown normalized to the respective benchmark sequence sample, which acts as internal standard; the TIR of a given RBS is divided by the TIR of the benchmark RBS run in the same plate.

and uncertainties of the predictions in the Learn phase of DBTL. The predictions of characteristics of interest for each potential variant can then be used in the Design phase.

Our overall experimental goal is to maximize the TIR by building and testing batches of RBS sequences with only a small number of DBTL cycle iterations. We demonstrate how machine learning (ML) algorithms can be incorporated into the DBTL cycle to predict (Learn) and recommend (Design) variants of a bacterial RBS with the goal of optimizing associated protein expression level in the specific genetic context (i.e., with specified bases upstream and downstream of the investigated RBS sequences). Our proposed machine learning guided DBTL cycle is summarized in Figure 1. Two types of ML algorithms are applied in the DBTL cycle. In the Learn phase, the goal is to train a predictor that will predict the protein expression level by learning from the logged data. We used a Bayesian, nonparametric regression algorithm, called Gaussian Process Regression (GPR)[28] in our pipeline. In addition to providing a predicted mean TIR, GPR also provides uncertainty estimates via its predicted standard deviation. GPR has been shown to perform well with small amounts of training data in biological prediction tasks.[28,29] The goal of the Design phase is to find a policy to recommend RBS sequences to query in batches so that we can identify the optimized RBS sequences within a given budget. We use a version of the Upper Confidence Bound multiarmed Bandit algorithms[29] to learn the policy. The policy uses the outputs from the GPR predictions to recommend useful batches for the Build and Test phases. We demonstrate that laboratory automation methods in the Build and Test phases result in high quality TIR data that is well suited for training machine learning methods in the subsequent Learn phase. The two types of algorithms cooperate with each other and provide powerful tools for DBTL cycle in genetic parts design.[29,30]

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

Using our proposed machine learning guided DBTL cycle (Figure 1), we were able to increase expression of our target protein by up to 35%, as compared to the very strong benchmark RBS.

2. RESULTS

We present our RBS-optimizing DBTL workflow that uses machine learning in Section 2.1. Machine learning is used in two different ways: (i) we show the efficacy of the ML recommendations in the Design stage (Section 2.2), and (ii) we demonstrate that the ML predictions are accurate in the Learn stage (Section 2.3). We present our new RBS sequence library in Section 2.4 and describe some interesting characteristics of the discovered sequences, as well as show the effectiveness of the automated laboratory workflow.

2.1. The Experimental Workflow. Our DBTL workflow, which uses machine learning to optimize protein expression, is shown in Figure 1. The Build and Test phases are driven chiefly by choices made by human researchers and the use of automated methods. Machine learning algorithms are applied in Learn and Design. In the Learn phase, we use the Gaussian Process regression algorithm to predict the TIR of RBS sequences comprising the experimental space. In the Design phase, we use the Bandit algorithm to recommend new RBS sequences based on the predictions from Learn.

The exact position of the RBS in the sequence upstream of the protein coding sequence (CDS) can be hard to pinpoint. However, most previous studies place the RBS in the 20 bp directly preceding the coding sequence. In our investigation, we are using an RBS that is known to have a very high TIR when expressing GFP and is present in the pBb series of plasmids.[31] This template RBS sequence is 20 base pairs (bp) long with the sequence TTTAAGA**AGGAGA**TATACAT, where the highlighted nucleotides constitute the core of the RBS. As this is the sequence against which new RBS sequences will be benchmarked, we will refer to this sequence as the *benchmark sequence* hereafter. Additionally, we have experimentally confirmed that modifying the core sequence is statistically more impactful on TIR than changes made outside of it (see Figure S5). This hypothesis has been built based on reported biases toward certain bases present in the core of the RBS but absent outside of it. For example, according to ref 32 there is a strong bias toward A and G bases in the core region of the RBS. Similarly, outside of the 6 bases of the core in the wider 20 bp context of the RBS there is no significant bias toward any particular base, which suggests that these bases do not contribute to the overall TIR of a given RBS. Focusing on the 6 to 8 bp core sequence is a common RBS design approach.[33] Hence, in our design, we focus on modification of the core at nucleotide positions −8 to −13 (relative to the start codon of the GFP; this is where the consensus Shine−Dalgarno AGGAGG sequence is usually found in wild type *E. coli*) of the RBS and we keep other positions the same as the benchmark sequence, i.e., TTTAAGA + NNNNNN + TATACAT, where N can be any nucleotide (A, C, G, T). The total experimental (variant) space is then $4^6 = 4096$.

In our genetic design, the investigated RBS controls the expression of green fluorescent protein (GFP). By controlling expression of a fluorescent protein with the RBS we can quickly assess the perceived relative TIR by measuring fluorescence of cells harboring the expression vector over time. Finally, the mRNA is transcribed from an IPTG-inducible promoter pLlacO-1. Inducible expression allows synchronization of the start of the GFP expression in all the cultures by addition of IPTG. Since standardization and comparative studies should be done in as similar genetic contexts as possible, the design of this device has been deliberately kept simple to make such studies easier.[34]

2.2. Design: Performance of the Recommendation Algorithm. The Bandit recommendations were made using the batch Upper Confidence Bound multiarmed Bandit algorithm.[29] In short, this algorithm is a stochastic method of probing the experimental space, and is sometimes referred to as Bayesian optimization. It maximizes the reward (output) from testing a limited number of instances from a large pool that cannot be tested exhaustively because of limited resources (time, computational power, money). It balances the exploration−exploitation paradigm, where exploration focuses on testing data points that maximize information gain and exploitation focuses on recommending RBSs with high predicted TIR.

Figure 2A shows the results for all the RBS groups tested experimentally. In each experimental round, in addition to the new RBS designs, we measure the TIR of the benchmark RBS as the internal standard. We then obtain the normalized TIR (called *TIR ratio*) by taking the ratio between the raw TIR of a new design and the average TIR of benchmark sequences in each round (which are run in triplicate in each round). Figure S6 shows these results in terms of raw TIRs.

To generate the data set from which the algorithm would learn, we decided to characterize a total of 450 RBS variants, a little over 10% of the whole experimental space. About a quarter of the designs are experimental controls to provide a baseline for comparison. To fit our automated workflow, we divided the 450 variants into batches of 90, split into 4 design rounds.

In the zeroth round we tested two batches of designs, giving a total of 180 variants split as follows:

- BPS-NC and BPS-C group: 60 RBS sequences that are subsequent single nucleotide variations of all 20 nucleotides of the original, benchmark sequence. This batch is designed to show us the influence of such single nucleotide changes on the overall performance of the RBS and the potential impact of changes made beyond the core part (see Supplementary Figure S5).
- UNI group: 30 RBS sequences that were uniformly randomized, i.e., equal probability of choosing any nucleotide for each position. This group shows the performance of RBSs generated randomly.
- PPM group: 30 RBS sequences randomized based on the position probability matrix (PPM) generated from all the naturally occurring RBS sequences in the *E. coli* genome.[35] This group shows the performance of RBSs generated randomly, but following the natural nucleotide distribution.
- Bandit-0: 60 RBS sequences recommended by our implementation of the recommendation algorithm based on a data set obtained from literature,[36] which contains 113 nonrepeated records for 56 unique RBS sequences with their respective TIRs. This data set has been used because of the perceived similarity of its goal to that of this work—prediction of TIR based on phenotypic output.

In the subsequent 3 rounds, with one batch each, all 90 designs were generated using our machine learning algorithm

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

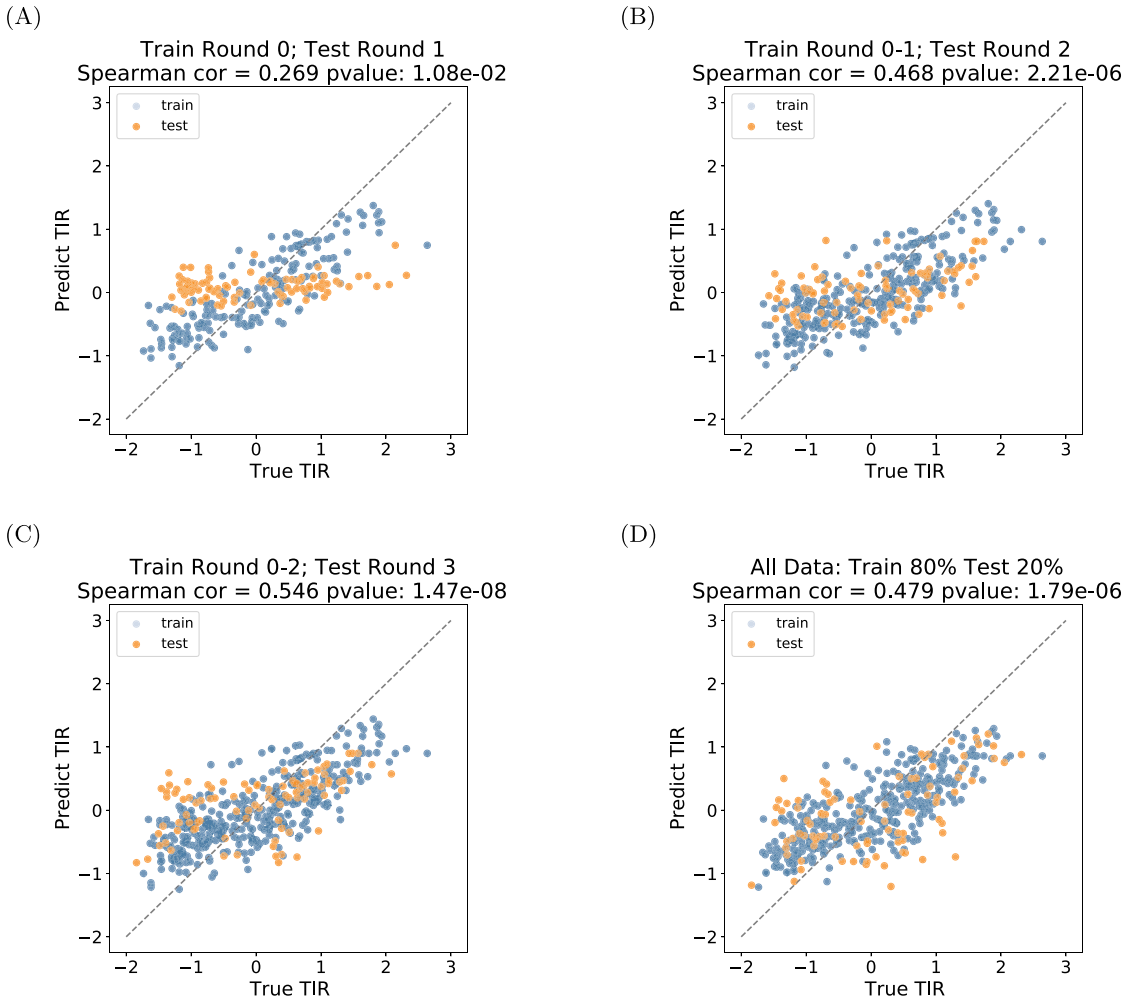


Figure 3. Performance of the prediction algorithm (no kernel normalization). The scatter plots A−D show the performance of the prediction algorithm calculated after each round. Note that the TIR values are normalized according to the standardization described in Section 4.2.1, which is different from the TIR ratio reported in Figure 2. The *x*-axis and *y*-axis are, respectively, the true measured TIR and the predicted TIR. In (A−C), we show at round $t = 1, 2, 3$, respectively, we train our predictor based on the previously obtained data (round 0 to $t - 1$) and show the predictions on both the training data (orange) and recommendations suggested by the Design phase (i.e., test data, blue). In (D), the predictor was trained on a randomly chosen 80% subset of all available data, and tested on the remaining 20% data. The Spearman correlation coefficient (with corresponding *p*-value; calculated using test data only) are provided in each plot's title on test data only. The *p*-value here is for the null hypothesis stating that two sets of data are uncorrelated, providing strong evidence that the predicted TIR are accurate.

based on the data obtained from the previous rounds (these groups are called Bandit 1−3, respectively).

All Round 0 groups (BPS-NC, BPS-C, UNI, PPM, Bandit-0) performed worse than our benchmark sequence in terms of TIR. The best-performing group was the BPS-NC, which is explained by the relatively small impact on the TIR of changes made outside the RBS core. The Bandit-0 group's performance is similar to randomly generated designs, despite being machine learning-driven, because of being trained on approximate data. Starting from Round 1, where the prediction and recommendation algorithms were fed data from Round 0, the results improved significantly, with a number of sequences performing better than the consensus Shine−Dalgarno sequence and in one case, better than the benchmark (by 8%). In Round 2 we observed further improvement by obtaining more sequences with TIRs similar to our benchmark sequence. Finally, in Round 3 the algorithm identified two sequences that were 34% and 15% stronger than the benchmark sequence.

In summary, 120 out of 450 sequences (BPS-NC, BPS-C, UNI, PPM) are experimental controls created by sequence randomization. Only a few of the random sequences were promising, but they were still 20% weaker than the benchmark sequence's TIR (Figure 3A). In fact, these 80% TIR ratio sequences were created by randomizing the sequence outside of the core RBS region (BPS-NC), which was statistically shown not to be significantly impactful on the TIR (note, this applies only to the 7 bps on either side of the core; see Supplementary Figure S1 and work by ref 33). Sequences from randomization of the core RBS (BPS-C, UNI, PPM) are more appropriate controls, from which the best sequence achieved only about 65% of the benchmark TIR. These results show that generating a strong RBS sequence by random mutation is a nontrivial task, when the tested data set is relatively small. In contrast, our Bandit-driven design gives much better results, with RBSs getting close to benchmark performance and even exceeding it.

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

Figure 2B shows the same results but divided into quantiles where the specific point for a given group shows the highest TIR for that quantile. The gradual increase for all quantiles can be observed for all Bandit groups, suggesting that the algorithms have a better understanding of the experimental space given more data. The decreased result in the 0.9th quantile compared to the maximum value for the Bandit 3 group can be attributed to the increased emphasis on exploitation that has been set for that round compared to others. We see this effect in Figure 2C (with details shown in Supplementary A.6 and Figure S1), where we colored the data points for Bandit 1−3 groups according to their relative exploration−exploitation affinity. Those with a high predicted mean are colored blue and represent exploitation, those colored red are with high predicted uncertainty and represent exploration. The implication of the fact that an RBS sequence chosen at random will have low TIR (as shown by UNI and PPM) is that most of the exploration will result in low TIRs. These results confirm that RBSs with high TIRs tend to come from exploitation of the design space, whereas the exploration points give relatively low TIRs. Note that the exploration is necessary to expand our knowledge to the unknown parts of the design space and in effect allow us to exploit it better.

Figure 2D shows the TIRs of RBSs tested in the Bandit groups divided into bins with width equal to a TIR ratio of 0.1. KDE plots have been overlaid to depict the calculated density for each group. The increase in prevalence of later Bandit groups in the higher bins is evident, especially for Bandit 2 and 3, constituting the bulk of results in the >0.8 TIR ratio bins. Notably, the distributions calculated for all the groups are bimodal—we discuss the possible reasons for that later in the text.

In Figure 2E we show a t-distributed stochastic neighbor embedding (t-SNE)[37] plot depicting the experimental space. Each RBS is located on the plot according to its distance from other RBSs as calculated by our weighted degree kernel with shift (see Section 4.2.2). The RBSs recommended by Bandit groups cover the majority of the design space. Additionally, a number of clusters were especially targeted by the recommendation algorithm. For example, the circled clusters labeled as "G-Rich Clusters" have been actively recommended by the algorithm. More specifically, sequences with 4 or more guanines in any position constituted 10% of the randomly selected sequences and 5, 9, 16, and finally 25% in each of the 4 Bandit guided batches, respectively.

2.3. Learn: Prediction of RBS Performance. In the Learn phase, we use a popular regression algorithm, Gaussian Process Regression (GPR). GPR has two benefits: (1) it provides the predicted TIR and the confidence interval needed for the UCB algorithm in the Design phase, and (2) it provides a natural way to calculate similarities between genomic sequences via kernels. Figure 3 shows how our implementation of the Gaussian Process algorithm performed in terms of predictions (on a hold out test set) in each round. As expected, the use of approximate data led to poor predictions in Round 0. The predictions improved for the subsequent rounds, and the Spearman correlation coefficient rose from 0.269 for Round 0 to 0.546 for Round 3.

We provide the evaluation of the Learn phase in terms of a ranking-based Spearman correlation coefficient, which has been shown to be a more suitable evaluation metric when the prediction is used for recommendation tasks than coefficient of determination (R^2) or Pearson correlation coefficient.[38,39] The regression task in the DBTL cycle is more challenging than the large-scale data-based regression tasks. In the early iterations, we have a limited number of data points and relatively high variability due to the measurement noise. However, since the predictions are only used for recommendations in the Design phase, instead of precise predictions of the mean for each RBS, we only need to provide a valid ranking for both the predicted mean TIR and uncertainties, and thus the ranking of the UCB scores. This is because, in the Design part, we select RBSs based on the ranks of the TIR predictions instead of numerical predictions. This would ensure that for each round we would be testing the required designs, even if we are not exactly accurate in terms of their numerical characteristics.

The performance of the Learn phase is also influenced by the exploration−exploitation balance in the Design phase. In each round, some data points were selected for exploration of the areas in which we have few or no tested data points. For those exploration points, since the predictor never learns their label distributions and patterns, there is no chance for the predictor to provide accurate predictions of TIR values, thus hurting the prediction performance in the short term. However, this is useful information for future predictions as it allows us to understand the whole underlying space instead of focusing on local suboptimal data points. In other words, we are intentionally sacrificing the accuracy of our predictions in each round to improve them in the future rounds. The effect of exploration of the space is the ability to find high TIR RBSs even with relatively low prediction performance.

2.4. Build and Test: Characteristics of the Tested Sequences. We present some important characteristics of the tested RBSs in Table 1, demonstrating the effectiveness of our

Table 1. Characteristics of the Library[a]

characteristics of the library	statistics
Total experimental space	4096
Planned constructs	450
Successfully constructed	445
Sequences with CV < 40%	79%
Sequences with CV < 20%	27%
Efficiency ratio of Bandit design (compared with random)	2
Raw TIR range	[4.93, 105.38]
TIR ratio range	[0.06, 1.34]
top RBS core	**TIR ratio**
GGGGGC	1.34
GGGGGT	1.15
GGCTAT	1.08
AGGAGA	1
GGCGTT	0.98
GGGGGG	0.98
GGCGAC	0.98
CAGGAG	0.96
GGCGAG	0.95
AGGAGG	0.39

[a]Top table presents some of the characteristics of our library. Bottom table presents 10 RBS sequences with their corresponding TIR ratios; the first 9 are the strongest sequences including the benchmark sequence (AGGAGA) and the last is the Shine−Dalgarno sequence (AGGAGG). CV is coefficient of variation (standard deviation (STD) of a sample divided by its mean; see details in Figures S11 and S12). Efficiency of the Bandit design is calculated by dividing the highest TIR found using machine learning by the highest TIR found using random sequence generation.

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

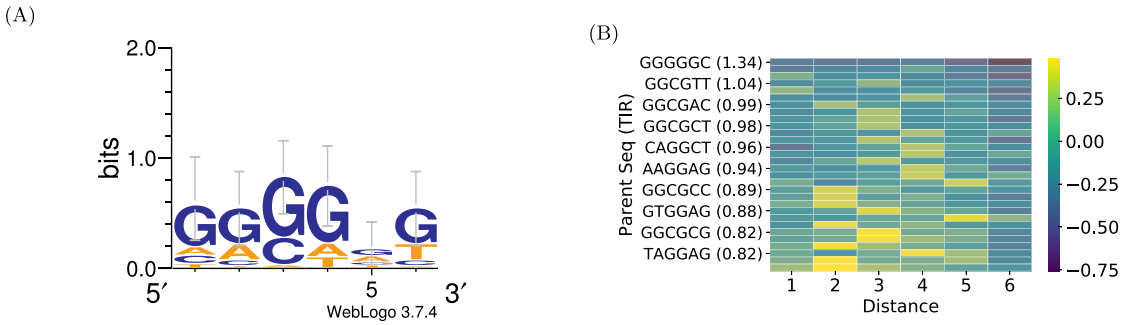


Figure 4. Characteristics of strong RBSs. (A) Sequence logo calculated for the Top 30 tested sequences. (B) Heatmap showing the edit (Hamming) distance required for positive change in TIR for RBSs with high and medium TIRs. The temperature scale shows the difference between a given RBS on the *y*-axis and the RBS with the strongest TIR at the given distance. Every second RBS is labeled for increased legibility.

Build and Test phases. Figure 4A shows the sequence logo calculated for the Top 30 sequences (Figure S9 shows the logo generated for all tested sequences). It is generally understood that guanine-rich sequences promote strong transcription. This expected bias toward guanine is clearly visible for all positions in our Top 30 RBSs. This result combined with the Bandit algorithm's bias toward the G-rich cluster shown in Figure 2D reinforces the notion that our algorithm successfully identified G-rich sequences as the ones with higher probability of having a high TIR value.

We discovered an interesting phenomenon of large edit distances between RBSs with high TIRs. In other words, given an RBS with high TIR, multiple sequence edits need to occur to reach another RBS with high TIR (we assume here that our data set includes most of the strongest RBSs for each distance−sequence combination). We define the edit distance as Hamming distance, that is, how many positions have to be changed to get from one sequence to the other (Hamming distance of 0 means that the sequences are identical and 6 means that they are two completely different sequences). Figure 4B shows the edit distance that is required for positive change in TIR for an RBS with TIR > 0.75. For RBSs with high TIRs (>1), the minimum distance that is required to increase the TIR is 2, with edit distances between 2 and 5 giving similar results. For RBSs with medium TIRs (<1), a distance of 1 is enough to produce a meaningful increase in TIR.

This means that as the TIRs of examined RBSs increase, exploring sequences that are increasingly dissimilar to the current candidates tends to give more meaningful improvement. As long as this does not impact targeted methods like machine learning-guided design, this implies that the low rate of natural mutations will be very slow to explore more dissimilar sequences on such a short distance,[40] which indicates that methods like Adaptive Laboratory Evolution may not be able to find very strong RBSs with a limited budget. In other words, because the examined sequence is relatively short (6 bp in a wider 20 bp context) the time required to accumulate 2 or more changes in the RBS region required for meaningful increase in TIR might be prohibitively long (Figure 2A shows comparison between efficiency of random versus directed sequence generation). In such cases, a directed process, like the one described here, should be strongly encouraged. This observation is in line with approaches seen in other disciplines, e.g., protein engineering, where more directed changes yield better results than random changes.[41]

Finally, while our strong sequences showed some affinity for the antisense sequence of the ribosome known to bind to RBS, they did not show any obvious secondary structures that could explain their TIRs (see Figure S10). This result combined with the unexpectedly bimodal nature of KDEs in Figure 2 reinforces the notion, based on the previously reported literature,[14,16] that there may be a number of different mechanisms governing the probability of effective RBS-ribosome binding.

3. DISCUSSION

In this work, we show how a machine learning guided approach and high-throughput, automated laboratory methods can be jointly applied to efficiently optimize a small genetic part, in this case maximizing the TIR of a bacterial RBS. In the Learn phase, we used Gaussian process regression to predict the TIR mean and uncertainty of that prediction. To represent RBSs and capture the similarities between them, we choose to use the Weighted Degree Kernel with Shift method, which fits well with Gaussian processes. In the Design phase we used an Upper Confidence Bound multiarmed Bandit algorithm to recommend sequences to be tested in batches. In the Build and Test phase, we performed our experiments using laboratory automation to increase their speed, reliability, and reproducibility. Using our proposed workflow and testing 450 RBS variants in 4 DBTL cycles, we designed and experimentally validated RBSs with high translation initiation rates equaling or exceeding the currently known strong RBSs in this genetic context by up to 34%. Furthermore, we have generated an extensive library of diverse RBSs that can be used as a basis for future studies. In the rest of the section, we first revisit our overall experimental goal and challenges encountered in each phase of the DBTL cycle (Figure 1). We then link our proposed methods with related work and discuss the potential generalization of our framework. We further discuss our design choices and open questions in this line of research. We refer to ref 42 for a more detailed discussion.

Our goal is to show the power of the machine learning guided DBTL cycle on RBS optimization. Our approach has shown that this combination of the two machine learning algorithms is able to correctly detect and exploit rules of biological design that otherwise require substantial time and experiments to uncover. We focus on the part-centric optimization,[27] which is an important task in synthetic biology. Understanding how to design the individual parts also allows us to extend the framework to better strain design,[23] where the part is optimized with a wider goal of strain optimization.

The machine learning guided DBTL framework has good potential to be generalized to multigene pathway design as

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

proposed by HamediRad, Chao et al.[43] and recently reviewed in ref 21. For example, the optimization goal can be adjusted to address combinatorial optimization for multigene and RBS scenarios; since this would be a large-scale data task, the current Gaussian Process regression prediction model could be updated to a deep Gaussian Process regression approach and the current Bandit algorithm could be optimized toward querying large design space to reduce computational complexity.[44]

In the Design phase, we focus on maximizing the TIR of RBSs, by gradually moving our emphasis from exploration to exploitation as we progress through the design rounds. While maximization could be the appropriate payoff function for optimizing RBSs or other small parts, other payoffs may be better in more complex cases. For example, when considering multigene metabolic engineering, maximizing expression of individual genes may result in excessive metabolic burden, which could be achieved within the bandit framework by combining different goals into a multiobjective method.[45] We constrain our design space over 6-core parts of RBS sequences in this study; the design space will increase exponentially when designing a longer sequence. The computational complexity of calculating and sorting acquisition functions (e.g., UCB scores) can be reduced by discretizing the space adaptively and hierarchically.[46,47]

In the Learn phase, our approach has correctly identified the correlation between high guanine content in the RBS and high TIR. We have achieved this despite the relatively low Spearman scores for our predictions. This observation of useful recommendations despite low prediction scores corroborates recent evidence from other studies.[20,48] Finally, our predictor Gaussian Process regression model, compared with previously described calculators using a deterministic thermodynamic approach, is able to show the uncertainty of the predictions, which can be used by our bandit algorithm to give better recommendations.

To further test the performance of our predictions and recommendations, we have generated a TIR label for each of the 4096 RBS sequences using the RBS calculator[12] and used our approach to simulate the search for the strongest designs in limited number of rounds. We show these results in Supplementary Figure S3, but in short, our approach was able to identify above 75% of the sequences stronger than the benchmark in up to 2 rounds and 99% of them in up to 3 rounds. This surprisingly high efficiency can be partially attributed to the source of the TIR predictions, which being generated algorithmically are easier to predict by our algorithm than real world data. Nevertheless, it shows that the algorithm can efficiently learn the sequence to TIR relationship and exploit it subsequently in a recommendation task.

In this study we have limited the number of design rounds to four. There were a number of reasons for this, including limitations on time and money, but also because the results obtained showed that we have achieved our goal of generating very strong RBS designs. There is a possibility that increasing the number of experimental rounds would enable us to improve the results further; however, this has to be put in the context of limited resources. For example, scanning the whole space would surely achieve the best results, i.e., would enable us to find the strongest possible RBS, but that would require unreasonable use of resources. Compared to solutions like the one reported by Hollerer,[27] our solution can be used when a high-volume method for data-generation is not available (for example, where there is no fluorescent readout available), while still providing the required results (optimized part).

There are still open questions that need to be addressed for the application of machine learning in synthetic biology. First, we would like to understand how we can extract more biologically important information from the decisions made by our algorithms. We have shown that the algorithms are able to exploit them, but it will be important to create tools that will enable their reliable extraction from the results obtained. Second, given the small number of RBS sequences tested, how can machine learning algorithms provide more accurate predictions and uncertainty quantification? Third, the generalizability of the method is unknown. We believe that the method described here would be useful for designing other small genetic parts, but the complexity of the task quickly increases with the size of the analyzed sequence, so the method's applicability might be impacted at some point. Similarly, the reusability of the obtained data set is currently unknown. The TIR of an RBS is dependent on its genetic context, but our Hamming distance and TIR impact of nucleotides outside of the core (Figures 4B and S5, respectively) studies indicate that as long as the changes in the genetic context are small, the obtained data set could serve as a basis for similar design efforts. For example, the data could be used to teach the algorithm for the first round of new designs, which in turn could decrease the number of rounds required to obtain the required characteristics. Finally, the practical optimal exploration−exploitation balance between rounds and samples is still an open question. In our work we have steered the recommendation process toward exploitation as we progressed through the rounds, in line with other works addressing this balance.[20]

We have found our approach of bringing machine learning and synthetic biology experts together very fruitful. Synthetic biology is promoting standardized and normalized testing in biology and naturally pairs with machine learning, which can leverage the high quality biological data sets generated when the correct design rules are observed. The addition of machine learning to synthetic biology also adds an additional layer of scrutiny to the generated data sets through the advanced statistical methods that can be used to design and analyze the experiments. On top of that, the use of automation has helped us to produce more reliable results, which gave us the required confidence in our predictions and recommendations. We envision that pairing machine learning with high-throughput automation will keep delivering a high number of good quality data sets and improved methods for biological engineering.

In the future we hope to extend the algorithm to other more complicated genetic elements, including promoters and terminators. However, it is important to reiterate that the experimental space grows exponentially with the number of examined positions, so the space becomes increasingly hard to cover with experiments, and so the percentage of the space that is measured will inevitably decrease, which might impact the expected recommendation results. To solve this problem, different algorithms or experimental techniques might be needed, but the general workflow can be reused.

4. MATERIALS AND METHODS

4.1. Laboratory Experimental Design. *4.1.1. Build: Construction of Genetic Devices. Plasmid Design.* The pBbB6c-GFP plasmid was used for all our designs. This plasmid contains the GFP mut3b CDS, expression of which

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

can be induced by the addition of isopropyl β-D-1-thiogalactopyranoside (IPTG) (Merck, Darmstadt, Germany, catalogue no. I5502). The original RBS for the GFP CDS was replaced using a combination of PCR and isothermal assembly. Primer sequences and the assembly strategy were generated using the Teselagen DESIGN software (Teselagen Biotechnology, San Francisco, CA).

PCR. PCR amplification of the cloning inserts was done using Q5 High-Fidelity 2X Master Mix (NEB, Ipswich, MA, catalogue no. M0492L). Twenty μL reactions were prepared by dispensing 1 μL of each 10 μM reverse primer into the wells of a 96-well PCR plate using the Echo liquid handler (Beckman Coulter, Brea, CA). A Mastermix consisting of polymerase premix, plasmid DNA template (pBbB6c, 5−10 ng per reaction), and the single 10 μM forward primer was prepared and dispensed using the FeliX liquid handler (Analytik Jena, Jena, Germany) or electronic multichannel pipet. Reactions were run using Touchdown PCR or standard PCR cycling methods in C1000 thermal cyclers (Bio-Rad, Hercules, CA). Capillary electrophoresis of PCR products was performed using the ZAG DNA Analyzer system (Agilent Technologies, Santa Clara, CA). Two μL of each PCR reaction was electrophoresed using the ZAG 130 dsDNA Kit (75−20 000 bp) or ZAG 110 dsDNA Kit (35−5000 bp) (Agilent Technologies, catalogue no. ZAG-110−5000; ZAG-130−5000). ZAG sample plates were prepared using the Sciclone G3 liquid handler (PerkinElmer, Waltham, MA). ProSize Data Analysis Software (Agilent Technologies) was used to generate gel images from the sample chromatograms, and amplicon sizes were estimated by reference to the upper and lower DNA markers spiked into each sample and a DNA ladder run in well H12 of each sample plate.

Isothermal DNA Assembly. Constructs were assembled using NEBuilder HiFi DNA Assembly Master Mix (NEB, catalogue no. E2621L). Reactions consisting of approximately equal amounts of the common fragment and the variable fragment were prepared using the FeliX liquid handler or electronic multichannel pipet, to a final volume of 5 or 10 μL. Assemblies were run in the thermal cycler for 1 h at 50 °C, followed by an infinite hold step at 4 °C. Finally, samples were incubated with of 50 nL of DpnI (NEB, catalogue no. R0176S) at 37 °C for 90 min to degrade any residual template DNA.

E. coli Transformation. The DH5α cell line (Thermo Fisher Scientific, Waltham, MA, catalogue no. 18265017) was made chemically competent using the Mix and Go *E. coli* Transformation Kit and Buffer Set (Zymo Research, Irvine, CA, catalogue no. T3001). Twenty μL of cells was aliquoted into each well of a cold 96-well PCR plate and stored at −80 °C for later use. Plates of cells were thawed on a −20 °C cold block before 3 μL of the assembly product was added and mixed using the FeliX liquid handler. Cells were incubated on a cold block for 2−5 min before being plated in a 96 (12 × 8) grid on Omnitrays containing LB (BD, Franklin Lakes, NJ, catalogue no. 244610) and 15 g/L agar (Merck, catalogue no. A1296) with 34 μg/mL chloramphenicol (Merck, catalogue no. C1919). Plates were incubated overnight at 37 °C. Cells were plated using the FeliX liquid handler.

Automated Colony Picking and Culturing. A PIXL colony picker (Singer Instruments, Roadwater, United Kingdom) was used to select individual colonies from the transformation plates using the 490−510 nm (cyan) light filter. Each selected colony was used to inoculate 1 mL of selective medium in a 2 mL square well 96 plate. They were then cultured overnight at 37 °C with shaking (300 rpm).

Glycerol Stock Preparation. 100 μL of sterile 80% (v/v) glycerol (Chem-Supply, Gillman, Australia, catalogue no. GA010) and 100 μL of overnight culture were combined in the wells of a 96 deep (2 mL) round well plate using the FeliX liquid handler or electronic multichannel pipet They were then sealed with a 96-well silicone sealing mat and transferred to a −80 °C freezer.

Sequencing. Strains that gave GFP fluorescence intensity readings similar to that of the original RBS were selected for sequence confirmation by capillary electrophoresis sequencing (CES) (Macrogen, Inc., Seoul, South Korea). The strains transformed with each of the selected constructs were grown to saturation in 5 mL LB medium with chloramphenicol selection (34 μg/mL). Plasmids were extracted from the cultures using the QIAprep Spin Miniprep Kit (QIAGEN, Hilden, Germany, catalogue no. 27106) according to the manufacturer's instructions. Plasmid concentrations were quantified using the Cytation 5 plate reader with the Take3Micro-Volume Plate (BioTek, Winooski, VT) and all fell in the range of 100−200 ng/μL. Samples of 20 μL of undiluted plasmid DNA were sequenced using a single primer (5′-CGATATAGGCGCCA-GCAA-3′) that binds approximately 150 bp upstream of the RBS. Reads were aligned with the template sequence in the Teselagen software.

4.1.2. Test: Culture Analysis. Test Strain Culture. Overnight cultures (six biological (restarted every time from glycerol stock) replicates for each batch) were started by inoculating 1 mL of LB medium supplemented with 34 μg/mL chloramphenicol with 2 μL of the glycerol stock in a 96 deep (2 mL) round well plate, using the FeliX liquid handler or electronic multichannel pipet. Cultures were incubated at 37 °C with shaking (300 rpm) for 17 h. The following morning, 20 μL of each overnight culture was added to 980 μL of fresh selection medium, and these cultures were grown at 37 °C with shaking in a 96 deep (2 mL) round well plate. After 90 min, cultures were induced with IPTG to a final concentration of 0.5 mM. This was done by transferring 1.0 μL of 0.1 M IPTG to each well of a flat-bottom clear polystyrene 96-well plate using the Echo liquid handler, then adding 300 μL of culture to each well using an electronic multichannel pipet.

Microplate Spectrophotometry. The plates were monitored in the Cytation 5 microplate reader immediately after the addition of IPTG. Cytation 5 acquisition and incubation/shaking settings were as follows: length of run: 8 h; interval: 10 min; continuous orbital shake at 237 rpm and slow orbital speed; excitation wavelength: 490/10 nm; emission wavelength: 515/10 nm; bottom read; gain: 60; read height: 7 mm; read speed: Sweep.

4.2. Machine Learning Experimental Design. Two types of machine learning algorithms have to be applied to drive the experimental design workflow shown in Figure 1. One type of machine learning algorithm is a prediction algorithm (Learn), which helps us learn the function of TIR with respect to RBS sequence. The other type of machine learning algorithm is a recommendation algorithm (Design), which recommends RBS sequences to query (Test) in each round (in the sense of a single pass through the DBTL cycle) based on the predictions from Learn.

In Round *t*, prediction and design are based on the results obtained in all previous rounds. Our implementation of the machine learning algorithms was tested in Python 3.6 and used

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

the scikit-learn library.[49] In the following paragraphs, we describe our machine learning pipeline (which is applicable to design Rounds 1−3, denoted as Bandit 1−3) in the following order: data preprocessing, prediction and kernels, and finally recommendation. The pipeline for our zeroth round of machine learning designs, denoted as Bandit-0, is reported in Supplementary A.4.

4.2.1. Data Preprocessing. For each RBS sequence, we measured the TIR of 6 biological replicates. We measure GFP fluorescence of the culture in the log phase as it is the most representative one. We also measure the Optical Density at 600 nm (OD600) of the culture every 10 min, which reflects how dense (how many cells) the culture is. Denote our measuring time as $t \in \{t_{\min}, ..., t_{\max}\}$, where $t_{\min}$ is the start and end of the log growth phase. At $t_{\min}$, the $\frac{\text{GFP}}{\text{OD}_{600}}$ value is at its minimum. We choose the time range to be 4 h, i.e., $t_{\max} - t_{\min} = 4$ h. In our experiment, TIR is calculated as an averaged sum of GFP fluorescence divided by OD600 of the culture over time (calculated using values from all the measured data points),

$$\text{TIR} = \frac{1}{t_{\max} - t_{\min}} \sum_{t=t_{\min}}^{t_{\max}} \frac{\text{GFP}(t)}{\text{OD}_{600}(t)}$$

For label preprocessing, we first adjust the TIR values in each round using the round-wise reference values. The reference value is the TIR of the benchmark sequence that is run in triplicate in each round. Specifically, in Round t, we subtract the TIR mean of the benchmark RBS measured in Round t from all TIR values measured in the same round, for each replicate separately. We then normalized the data by performing a logarithm transformation and standardization on the adjusted TIR label for each replicate separately. After normalization, each replicate has zero mean and unit variance. Furthermore, we also normalized the kernel matrix used for prediction by centring and unit-norm normalization, which is reviewed in detail in Supplementary A.2.1.

4.2.2. Prediction: Gaussian Process Regression with String Kernel. To find RBS sequences with the highest possible TIR score after a total number of rounds N, we consider our experimental design problem as a sequential optimization of an unknown reward function $f: \mathcal{D} \to \mathbb{R}$, where $\mathcal{D}$ is the set containing all RBS sequence points in the design space, and $f(\mathbf{x})$ is the TIR score of the 6-base core sequence of the RBS $\mathbf{x} \in \{A,C,G,T\}^6$. In each Round t, we choose a set of m points $\mathcal{S}_t \subset \mathcal{D}$ and observe the function value at each point in the selected set $\mathcal{S}_t$, i.e., $y_i = f(\mathbf{x}_i) + \epsilon$, for all $i \in \mathcal{S}_t$, where ϵ is the Gaussian random noise with unknown mean and standard deviation.

For the regression model, we have used a Bayesian nonparametric approach called *Gaussian Process Regression* (GPR).[28,30,50] We model f as a sample from a *Gaussian Process* $\mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$, which is specified by the mean function $\mu(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and the kernel function (also called *covariance function*) $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}') - \mu(\mathbf{x}'))]$. GPR can predict the posterior mean of the RBS sequences in the design space. At the same time, it can provide the posterior standard deviation, which represents the level of uncertainty for the prediction.

We review the predictions of GPR in Supplementary A.1.

The choice of kernel function is critical for accurate predictions, since it controls smoothness and amplitude of the function to be modeled. For Bandit designs in Round 0, since we only had access to a limited number of data points from the literature, we chose to use one of the basic string kernels, the *spectrum kernel*[51] to process the core 6 bp and dot product kernel[28] (with one-hot embedding) to process the 7 bp flanking sequences both upstream and downstream of the core sequence. Among various string kernels, which have been studied in the literature,[52] and we will introduce briefly in Supplementary A.2, we chose the one with the best reported performance, the *weighted degree kernel with shift* (WDS),[52,53] to represent the RBS sequences in subsequent rounds and specify the kernel function of GP. WDS is a type of a string kernel, which takes two sequences as inputs and outputs a scalar value that represents the similarity between the two sequences. WDS kernel does this by counting the matches of substrings of a certain length (i.e., kmers) that constitute the sequence. The maximum substring length is specified by l. The WDS takes into account the positional information by counting substrings starting from different positions, where the start position is specified by l. Additionally, the WDS kernel considers the shifting of substrings, with the maximum shift specified by s. This could be useful when there is a shift between two sequences. For example, two sequences **ACCTGA** and **CCTGA**A are in 1-shift.

We now define WDS kernel. Let $\mathbb{I}(A)$ be the indicator function, which equals 1 if A is true and 0 otherwise. Then $\mathbb{I}(\mathbf{x}_{[l+s:l+s+d]} = \mathbf{x}'_{[l:l+d]})$ indicates whether two substrings of length d match, between $\mathbf{x}$ starting from position $l + s$ and $\mathbf{x}'$ starting from position l. This is similarly done for $\mathbb{I}(\mathbf{x}_{[l:l+d]} = \mathbf{x}'_{[l+s:l+s+d]})$. By having these two terms considering substrings of two sequences with starting positions differing by s characters, the WDS can measure shifted positional information. When $s = 0$, the kernel function counts the matches with no shift between sequences. Let $\mathbf{x}, \mathbf{x}'$ be two RBS sequences with length L; the WDS kernel is defined as

$$k_l^{\text{WDS}}(\mathbf{x}, \mathbf{x}') = \sum_{d=1}^{l} \beta_d \sum_{l=1}^{L-d+1} \gamma_l \sum_{s=0, s+l \le L}^{S(l)} \delta_s \left(\mathbb{I}(\mathbf{x}_{[l+s:l+s+d]} = \mathbf{x}'_{[l:l+d]}) + \mathbb{I}(\mathbf{x}_{[l:l+d]} = \mathbf{x}'_{[l+s:l+s+d]})\right) \quad (1)$$

where $\beta_d = \frac{2(l-d+1)}{l(l+1)}$, $\delta_s = \frac{1}{2(s+1)}$, and γ_l is a weighting parameter over the position in the sequence, where we chose to use a uniform weighting over the sequences, i.e., $\gamma_l = 1/L$ ($L = 20$ in our design).

$S(l)$ determines the maximum shift at position l. Furthermore, we normalize the kernel with centring and unit-norm in terms of all sequences in the design space.

The hyperparameters for kernel, including maximum substring length $l = 6$, maximum shift length $S(l) = 1$, and the standard deviation of GP noise $\alpha = 1$ were chosen on the basi of 10-repeat 5-fold cross validation. The values of hyperparameters considered were $l \in \{3, 4, 5, 6\}$, $S(l) = \{0, 1, 2\}$, $\alpha = [0.5, 1, 2]$.

4.2.3. Recommendation: Upper Confidence Bound Multi-armed Bandit Algorithm. To recommend the RBS sequences to query in the next round, we have used the *Upper Confidence Bound* (UCB) batch algorithm,[9] which provides *exploration−exploitation balance*. On one hand, UCB exploits the function

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

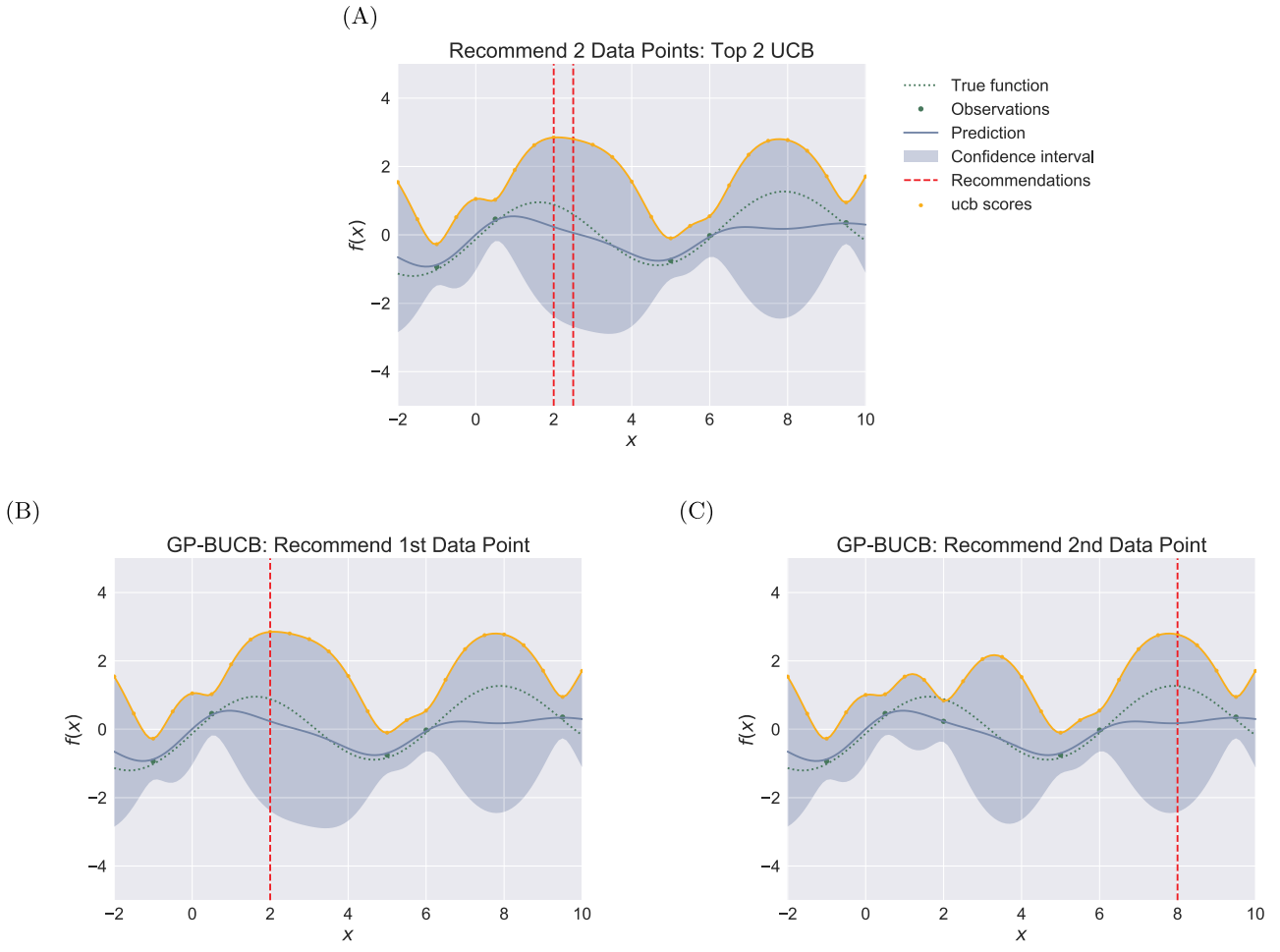


Figure 5. Batch recommendation illustration. We use the batch size of 2, with 5 initial observations. The design space is 24 uniformly distributed points in the range [−2, 10], i.e., −2, −1.5, −1, ..., 9.5, 10. The confidence intervals are shown with predicted mean ±1.96 standard deviation. (A) Top UCB recommendations. The recommendations are 2 data points with top UCB scores, chosen with GP predictions. (B,C) Batch UCB recommendations. Panel (B) shows the first recommended sequence, (C) shows the new predicted confidence interval and the second recommendation based on that updated interval.

in terms of the design space, that is to pinpoint sequences that are believed to have high labels (i.e., high predicted mean); on the other hand, UCB explores the design space where we have little information and sequences have a chance to have high labels (i.e., high predicted standard deviation). More precisely, the UCB algorithm selects RBS sequences $\mathbf{x}_i \in \mathcal{D}$ with the maximum upper confidence bound at Round t, i.e.,

$$\text{argmax}_{\mathbf{x}_i \in \mathcal{D}}(\mu_{t-1}(\mathbf{x}_i) + \beta_t \sigma_{t-1}(\mathbf{x}_i)) \quad (2)$$

where β_t is a hyperparameter balancing the exploration and exploitation, $\mu_t(\mathbf{x}_i)$, $\sigma_t(\mathbf{x}_i)$ are the predicted mean and standard deviation (that is uncertainty of the prediction) at Round t for the sequence $\mathbf{x}_i$. We call $\mu_{t-1}(\mathbf{x}_i) + \beta_t \sigma_{t-1}(\mathbf{x}_i)$ the *UCB score* of sequence $\mathbf{x}_i$ at Round $t - 1$.

Since experimentally labeling sequences is time-consuming, it is unrealistic to recommend sequences sequentially (i.e., one-by-one) and then wait for the label to be tested and used to improve the model. Instead, we can recommend RBS sequences in a batch of size m. However, this approach may end up recommending similar sequences in the same local maximum (e.g., $x = 2$, $x = 2.5$ in this example). Since GPR assumes similar sequences would have similar labels (e.g., by knowing $x = 2$ we can gain information about $x = 2.5$ as well), we prefer to not waste time and money on labeling sequences with high similarities in the same batch. To counter this, we use a batch recommendation strategy that is designed to avoid recommending such highly similar sequences in the same batch, described below.

A key property of Gaussian Process regression is that the predicted standard deviation depends only on features, not on the labels. One can make use of this property to design a batch upper confidence bound (BUCB) algorithm.[29] The strategy here is to recommend RBS sequences one-by-one by sequentially adding the newly recommended RBSs' predicted means (without testing them) into the training data and updating UCB scores.

As illustrated in Figure 5B, the algorithm recommends the data point with maximum UCB score based on the predictions over the initial 5 observations. We then add the recommended data point ($x = 2$) into the training data set with the predicted mean of that point as the label (note it is not the true label), and update the predicted standard deviation, then we finally update the UCB scores. The second data point is then recommended based on the new UCB scores. Figure 5C shows that since we assume we have observed $x = 2$, the new predicted standard deviation of the data points in design space around $x = 2$ decreases, so instead of recommending a similar

https://doi.org/10.1021/acssynbio.2c00015
ACS Synth. Biol. 2022, 11, 2314−2326

data point $x = 2.5$, the algorithm recommends $x = 8$, which is in another local maximum design area. In this way, the batch recommendation can potentially cover more local maximum areas than the sequential design. In summary, the exploration efficiency is improved since the recommended sequences in one batch will tend to be in different design areas so that the information gain is maximized. In our design, we have set the batch size to 90, to fit the experimental batch. Finally, we set $\beta_t = 2$ for Round 0−3 and $\beta_t = 0$ for the last round to allow for more exploitation.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acssynbio.2c00015.

Sequences of plasmids and oligos and assembly reports used in this study (ZIP)

A detailed text description of the machine learning methods, including Gaussian Process regression, choices of kernels, batch recommendation, experimental settings, additional results, and model evaluation on RBS calculator data; Figures S1−S12; Tables S1−S2 (PDF)

■ AUTHOR INFORMATION

Corresponding Author

Cheng Soon Ong − *Machine Learning and Artificial Intelligence Future Science Platform, CSIRO, Canberra, ACT 2601, Australia; Department of Computer Science, Australian National University, Canberra, ACT 2601, Australia; Data61, CSIRO, Canberra, ACT 2601, Australia*; Email: cheng-soon.ong@data61.csiro.au

Authors

Mengyan Zhang − *Machine Learning and Artificial Intelligence Future Science Platform, CSIRO, Canberra, ACT 2601, Australia; Department of Computer Science, Australian National University, Canberra, ACT 2601, Australia; Data61, CSIRO, Canberra, ACT 2601, Australia*; orcid.org/0000-0003-4853-5066

Maciej Bartosz Holowko − *Synthetic Biology Future Science Platform and Land and Water, CSIRO, Canberra, ACT 2601, Australia*; orcid.org/0000-0002-1535-4296

Huw Hayman Zumpe − *Synthetic Biology Future Science Platform and Land and Water, CSIRO, Canberra, ACT 2601, Australia*

Complete contact information is available at: https://pubs.acs.org/10.1021/acssynbio.2c00015

Author Contributions

M. Zhang and C. S. Ong designed and implemented the machine learning algorithms and workflow. M. B. Holowko and H. Hayman Zumpe have designed and performed the laboratory experiments. M. B. Holowko and C. S. Ong conceived and planned the project. All authors analyzed the data, contributed to and reviewed the manuscript.

Notes

The authors declare no competing financial interest.

All code and data required to reproduce the results are available at Github (San Francisco, CA): https://github.com/mholowko/Solaris/tree/master/synbio_rbs. All the processed and raw data are included in the repository. The pBbB6c plasmid is available on Addgene (Watertown, MA). Other strains and plasmids are available on request from the authors; MTAs will need to be negotiated between the parties.

■ ACKNOWLEDGMENTS

The authors would like to acknowledge CSIRO's Machine Learning and Artificial Intelligence, and Synthetic Biology Future Science Platforms for providing funding for this research. The authors would also like to thank CSIRO BioFoundry for help with performing the experiments.

■ REFERENCES

(1) Brophy, J. A. N.; Voigt, C. A. Principles of genetic circuit design. *Nat. Methods* **2014**, *11*, 508−520.

(2) Canton, B.; Labno, A.; Endy, D. Refinement and standardization of synthetic biological parts and devices. *Nat. Biotechnol.* **2008**, *26*, 787−793.

(3) Stanton, B. C.; Nielsen, A. A. K.; Tamsir, A.; Clancy, K.; Peterson, T.; Voigt, C. A. Genomic mining of prokaryotic repressors for orthogonal logic gates. *Nat. Chem. Biol.* **2014**, *10*, 99−105.

(4) Yeoh, J. W.; Ng, K. B. I.; Teh, A. Y.; Zhang, J.; Chee, W. K. D.; Poh, C. L. An Automated Biomodel Selection System (BMSS) for Gene Circuit Designs. *ACS Synth. Biol.* **2019**, *8*, 1484−1497.

(5) Nielsen, A. A. K.; Der, B. S.; Shin, J.; Vaidyanathan, P.; Paralanov, V.; Strychalski, E. A.; Ross, D.; Densmore, D.; Voigt, C. A. Genetic circuit design automation. *Science* **2016**, *352*, aac7341.

(6) Xia, T.; SantaLucia, J.; Burkard, M. E.; Kierzek, R.; Schroeder, S. J.; Jiao, X.; Cox, C.; Turner, D. H. Thermodynamic Parameters for an Expanded Nearest-Neighbor Model for Formation of RNA Duplexes with Watson-Crick Base Pairs. *Biochemistry* **1998**, *37*, 14719−14735.

(7) Chen, Y.-J.; Liu, P.; Nielsen, A. A. K.; Brophy, J. A. N.; Clancy, K.; Peterson, T.; Voigt, C. A. Characterization of 582 natural and synthetic terminators and quantification of their design constraints. *Nat. Methods* **2013**, *10*, 659−664.

(8) Reeve, B.; Hargest, T.; Gilbert, C.; Ellis, T. Predicting translation initiation rates for designing synthetic biology. *Front. Bioeng. Biotechnol.* **2014**, *2*, 1.

(9) Lattimore, T.; Szepesvári, C. *Bandit Algorithms*; Cambridge University Press, 2020.

(10) Seo, S. W.; Yang, J.-S.; Kim, I.; Yang, J.; Min, B. E.; Kim, S.; Jung, G. Y. Predictive design of mRNA translation initiation region to control prokaryotic translation effciency. *Metabolic631 Engineering* **2013**, *15*, 67−74.

(11) Na, D.; Lee, D. RBSDesigner: software for designing synthetic ribosome binding sites that yields a desired level of protein expression. *Bioinformatics* **2010**, *26*, 2633−2634.

(12) Salis, H. M.; Mirsky, E. A.; Voigt, C. A. Automated design of synthetic ribosome binding sites to control protein expression. *Nat. Biotechnol.* **2009**, *27*, 946−950.

(13) Reis, A. C.; Salis, H. M. An Automated Model Test System for Systematic Development and Improvement of Gene Expression Models. *ACS Synth. Biol.* **2020**, *9*, 3145−3156.

(14) Espah Borujeni, A.; Salis, H. M. Translation Initiation is Controlled by RNA Folding Kinetics via a Ribosome Drafting Mechanism. *J. Am. Chem. Soc.* **2016**, *138*, 7016−7023.

(15) Goss, P. J. E.; Peccoud, J. Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. *Proc. Natl. Acad. Sci. U. S. A.* **1998**, *95*, 6750−6755.

(16) Saito, K.; Green, R.; Buskirk, A. R Translational initiation in E. coli occurs at the correct sites genome-wide in the absence of mRNA-rRNA base-pairing. *eLife* **2020**, *9*, No. e55002.

(17) Scherer, G. F.; Walkinshaw, M. D.; Arnott, S.; Morré, D. The ribosome binding sites recognized by E. coli ribosomes have regions with signal character in both the leader and protein coding segments. *Nucleic Acids Res.* **1980**, *8*, 3895−3908.

(18) de Smit, M. H.; van Duin, J. Translational initiation on structured messengers: Another role for the shine-dalgarno interaction. *J. Mol. Biol.* **1994**, *235*, 173−184.

(19) Camacho, D. M.; Collins, K. M.; Powers, R. K.; Costello, J. C.; Collins, J. J. Next-Generation Machine Learning for Biological Networks. *Cell* **2018**, *173*, 1581−1592.
(20) Radivojevic, T.; Costello, Z.; Workman, K.; Garcia Martin, H. A machine learning Automated Recommendation Tool for synthetic biology. *Nat. Commun.* **2020**, *11*, 4879.
(21) Lawson, C. E.; Martí, J. M.; Radivojevic, T.; Jonnalagadda, S. V. R.; Gentz, R.; Hillson, N. J.; Peisert, S.; Kim, J.; Simmons, B. A.; Petzold, C. J.; Singer, S. W.; Mukhopadhyay, A.; Tanjore, D.; Dunn, J. G.; Garcia Martin, H. Machine learning for metabolic engineering: A review. *Metabol. Eng.* **2021**, *63*, 34−60.
(22) Freemont, P. S. Synthetic biology industry: data-driven design is creating new opportunities in biotechnology. *Emerging Topics in Life Sciences* **2019**, *3*, 651−657.
(23) Jervis, A. J.; Carbonell, P.; Vinaixa, M.; Dunstan, M. S.; Hollywood, K. A.; Robinson, C. J.; Rattray, N. J. W.; Yan, C.; Swainston, N.; Currin, A.; Sung, R.; Toogood, H.; Taylor, S.; Faulon, J.-L.; Breitling, R.; Takano, E.; Scrutton, N. S. Machine Learning of Designed Translational Control Allows Predictive Pathway Optimization in Escherichia coli. *ACS Synth. Biol.* **2019**, *8*, 127−136.
(24) Costello, Z.; Martin, H. G. A machine learning approach to predict metabolic pathway dynamics from time-series multiomics data. *npj Syst. Biol. Appl.* **2018**, *4*, 19.
(25) Alipanahi, B.; Delong, A.; Weirauch, M. T.; Frey, B. J. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat. Biotechnol.* **2015**, *33*, 831−838.
(26) Angermueller, C.; Pärnamaa, T.; Parts, L.; Stegle, O. Deep learning for computational biology. *Molecular Systems Biology* **2016**, *12*, 878.
(27) Höllerer, S.; Papaxanthos, L.; Cathrin Gumpinger, A.; Fischer, K.; Beisel, C.; Borgwardt, K.; Benenson, Y.; Jeschek, M. Large-scale DNA-based phenotypic recording and deep learning enable highly accurate sequence-function mapping. *Nat. Commun.* **2020**, DOI: 10.1038/s41467-020-17222-4.
(28) Rasmussen, C. E. In *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2−14, 2003, Tübingen, Germany, August 4−16, 2003, Revised Lectures*; Bousquet, O., von Luxburg, U., Rätsch, G., Eds.; Springer Berlin Heidelberg: Berlin, Heidelberg, 2004; pp 63−71.
(29) Desautels, T.; Krause, A.; Burdick, J. W. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *J. Mach. Learn. Res.* **2014**, *15*, 3873−3923.
(30) Srinivas, N.; Krause, A.; Kakade, S. M.; Seeger, M. W. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory* **2012**, *58*, 3250−3265.
(31) Lee, T. S.; Krupa, R. A.; Zhang, F.; Hajimorad, M.; Holtz, W. J.; Prasad, N.; Lee, S. K.; Keasling, J. D. BglBrick vectors and datasheets: A synthetic biology platform for gene expression. *J. Biol. Eng.* **2011**, *5*, 12.
(32) Shultzaberger, R. K.; Bucheimer, R.; Rudd, K. E.; Schneider, T. D. Anatomy of Escherichia coli ribosome binding sites. *J. Mol. Biol.* **2001**, *313*, 215−228.
(33) Jeschek, M.; Gerngross, D.; Panke, S. Rationally reduced libraries for combinatorial pathway optimization minimizing experimental effort. *Nat. Commun.* **2016**, *7*, 11163.
(34) Beal, J.; Baldwin, G. S.; Farny, N. G.; Gershater, M.; Haddock-Angelli, T.; Buckley-Taylor, R.; Dwijayanti, A.; Kiga, D.; Lizarazo, M.; Marken, J.; de Mora, K.; Rettberg, R.; Sanchania, V.; Selvarajah, V.; Sison, A.; Storch, M.; Workman, C. T. Comparative analysis of three studies measuring fluorescence from engineered bacterial genetic constructs. *PLoS One* **2021**, *16*, 1−15.
(35) Barrick, D.; Villanueba, K.; Childs, J.; Kalil, R.; Schneider, T. D.; Lawrence, C. E.; Gold, L.; Stormo, G. D. Quantitative analysis of ribosome binding sites in E. coli. *Nucleic Acids Res.* **1994**, *22*, 1287−1295.
(36) Jervis, A. J.; Carbonell, P.; Vinaixa, M.; Dunstan, M. S.; Hollywood, K. A.; Robinson, C. J.; Rattray, N. J.; Yan, C.; Swainston, N.; Currin, A.; et al. Machine learning of designed translational control allows predictive pathway optimization in Escherichia coli. *ACS synthetic biology* **2019**, *8*, 127−136.
(37) van der Maaten, L.; Hinton, G. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579−2605.
(38) Schober, P.; Boer, C.; Schwarte, L. A. Correlation Coefficients: Appropriate Use and Interpretation. *Anesth. Analg.* **2018**, *126*, 1763.
(39) Kang, J.-S.; Shin, D.-H.; Baek, J.-W.; Chung, K. Activity Recommendation Model Using Rank Correlation for Chronic Stress Management. *Appl. Sci.* **2019**, *9*, 4284.
(40) Lee, H.; Popodi, E.; Tang, H.; Foster, P. L. Rate and molecular spectrum of spontaneous mutations in the bacterium Escherichia coli as determined by whole-genome sequencing. *Proc. Natl. Acad. Sci. U. S. A.* **2012**, *109*, E2774−E2783.
(41) Jäckel, C.; Kast, P.; Hilvert, D. Protein Design by Directed Evolution. *Annu. Rev. Biophys.* **2008**, *37*, 153−173.
(42) Zhang, M.; Ong, C. S. Opportunities and Challenges in Designing Genomic Sequences. In *ICML Workshop on Computational Biology*; International Conference on Machine Learning, 2021.
(43) HamediRad, M.; Chao, R.; Weisberg, S.; Lian, J.; Sinha, S.; Zhao, H. Towards a fully automated algorithm driven platform for biosystems design. *Nat. Commun.* **2019**, *10*, 1−10.
(44) Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R. P.; de Freitas, N. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE* **2016**, *104*, 148−175.
(45) Shu, L.; Jiang, P.; Shao, X.; Wang, Y. A new multi-objective Bayesian optimization formulation with the acquisition function for convergence and diversity. *J. Mech. Design* **2020**, *142*, 091703.
(46) Shekhar, S.; Javidi, T.; et al. Gaussian process bandits with adaptive discretization. *Electron. J. Statist.* **2018**, *12*, 3829−3874.
(47) Zhang, M.; Tsuchida, R.; Ong, C. S. Gaussian Process Bandits with Aggregated Feedback. In *36th AAAI Conference on Artificial Intelligence*; Association for the Advancement of Artificial Intelligence, 2022.
(48) Opgenorth, P.; Costello, Z.; Okada, T.; Goyal, G.; Chen, Y.; Gin, J.; Benites, V.; de Raad, M.; Northen, T. R.; Deng, K.; Deutsch, S.; Baidoo, E. E. K.; Petzold, C. J.; Hillson, N. J.; Garcia Martin, H.; Beller, H. R. Lessons from Two Design−Build−Test−Learn Cycles of Dodecanol Production in Escherichia coli Aided by Machine Learning. *ACS Synth. Biol.* **2019**, *8*, 1337−1351.
(49) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825−2830.
(50) Romero, P. A.; Krause, A.; Arnold, F. H. Navigating the protein fitness landscape with Gaussian processes. *Proc. Natl. Acad. Sci. U. S. A.* **2013**, *110*, E193−E201.
(51) Leslie, C.; Eskin, E.; Noble, W. S. In *Biocomputing 2002*; World Scientific, 2001; pp 564−575.
(52) Ben-Hur, A.; Ong, C. S.; Sonnenburg, S.; Schölkopf, B.; Rätsch, G. Support Vector Machines and Kernels for Computational Biology. *PLoS Comput. Biol.* **2008**, *4*, 1−10.
(53) Rätsch, G.; Sonnenburg, S.; Schölkopf, B. RASE: recognition of alternatively spliced exons in C.elegans. *Bioinformatics* **2005**, *21* (Suppl 1), i369−i377.