A primer on model-guided exploration of fitness landscapes for biological sequence design

Sam Sinai[1,2,3] and Eric D. Kelsic[1, 2]

Abstract

Machine learning methods are increasingly employed to address challenges faced by biologists. One area that will greatly benefit from this cross-pollination is the problem of biological sequence design, which has massive potential for therapeutic applications. However, significant inefficiencies remain in communication between these fields which result in biologists finding the progress in machine learning inaccessible, and hinder machine learning scientists from contributing to impactful problems in bioengineering. Sequence design can be seen as a search process on a discrete, high-dimensional space, where each sequence is associated with a function. This sequence-to-function map is known as a Fitness Landscape [1, 2]. Designing a sequence with a particular function is hence a matter of discovering such a (often rare) sequence within this space [3]. Today we can build predictive models with good interpolation ability due to impressive progress in the synthesis and testing of biological sequences in large numbers, which enables model training and validation. However, it often remains a challenge to find useful sequences with the properties that we like using these models. In particular, in this primer we highlight that algorithms for experimental design, what we call exploration strategies, are a related, yet distinct problem from building good models of sequence-to-function maps. We review advances and insights from current literature -by no means a complete treatment- while highlighting desirable features of optimal model-guided exploration, and cover potential pitfalls drawn from our own experience. This primer can serve as a starting point for researchers from different domains that are interested in the problem of searching a sequence space with a model, but are perhaps unaware of approaches that originate outside their field.

Correspondence: sam.sinai@dynotx.com, eric.kelsic@dynotx.com
1:Dyno Therapeutics, Cambridge, MA 2: Wyss Institute For Biologically Inspired Engineering at Harvard Medical School, Boston, MA 3: Harvard University, Cambridge, MA

Contents

1. Preliminaries

While we attempt to cover the advances that capture methods in approaching sequence design (or adjacent) problems, this is not a review and the field would always move faster than we can update this document. We often find ourselves explaining different facets of the challenge to researchers from different fields, and hence we think it might be useful as a resource to share and lower the barrier for working on this problem, or perhaps help in mak-

ing known approaches more accessible. As the purpose of this writing is to enable better algorithms, we focus on methods (rather than results) when discussing approaches that were attempted to-date.

2. What is the problem?

A "fitness landscape" is defined as a map $\Phi : \mathbf{x} \rightarrow \mathbf{y}$ between biological sequences $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ and their fitnesses $\mathbf{Y} = \{\mathbf{y}_1, \cdots, \mathbf{y}_n\}$[1]. Sequences $\mathbf{X}$ are each made up of alphabet (residues) $\Sigma = \{\sigma_1, \cdots, \sigma_k\}$ [1]. Here, the term *fitness* captures the biological desirability of a sequence. In nature, this can refer to the effect of multiple functionalities $\mathbf{y} = \{y_1, \cdots, y_m\}$ together on the organism's ability to survive or reproduce. For engineers, it might refer to other desired properties that don't necessarily align with biological fitness. The challenge for evolution or bioengineers is to find sequences $\mathbf{x}$ that exhibit some desired profile of functionalities $\mathbf{y}^*$ [3]. The purpose of the primer is to introduce the reader to exploration algorithms $\mathcal{E}$ that query Φ as oracles to find sequences with the desired properties $\mathbf{y}^*$. Given that we do not know if sequences we are looking for actually exist, this may be an impossible task. However, assuming such profile exists, we aim to propose $\mathcal{E}$ that can find solutions within some acceptable distance of $\mathbf{y}^*$ with high probability.

In order to achieve this objective, it is often helpful to build a *model* of the landscape, Φ' because accessing Φ directly is rather costly. Even evolutionary processes have an implicit model of fitness landscapes: They assume that the current members of the population are the most promising candidates to be modified towards a better sequence, i.e. there is correlation between nearby sequences and their function. Engineers can build explicit models of

---

[1]$|\Sigma| = 4$ for DNA and RNA nucleotides, and $|\Sigma| = 20$ for standard amino-acids which are the building blocks of proteins

the landscape Φ', which we will discuss in section 3.

In its simplest form the challenge would be to find a single sequence that satisfies a single criterion. For example, the green fluorescent protein (GFP) is a 238 amino-acid protein that is a good experimental test-case for machine-guided modeling and optimisation [4, 5, 6, 7, 8, 9], and one may look to improve its fluorescence ($y \in \mathbb{R}$), subject to the constraints that the protein still successfully folds.

2.1 Why are biologists interested in building models of the landscape?

While optimising a protein like GFP my not be very impactful *per se*, as it is pretty good at what it does already, there are other natural proteins that perform the tasks we are interested in, but we would greatly benefit from optimising them. For instance, capsid engineering [10], a domain that we work on, stands to benefits massively from improved capsids that enable delivery of genes into specific tissues in order to treat currently untreatable genetic diseases. While the known natural variants of these capsid proteins show the desired traits, their current capabilities are far below the levels that make them appealing as therapeutics.

Furthermore, as these landscapes are extremely large, we must understand a landscape's structure by sampling a very small portion of it. Evolutionary biologists are particularly interested in understanding landscape structure, because it can help make the process of evolution more predictable [11, 12, 13]. Such understanding has theoretical as well as practical implications. Practically, it can help us in estimating the probability of developing drug resistance for certain pathogens better, or even estimating the chances of a virus jumping from animal hosts to humans. More generally the structure of the landscape determines how hard it is for evolution to "discover"

4

novel functions. These can help us progress towards understanding big scientific questions. For instance, we know that life on earth eventually discovered oxidative processes as a means of producing energy. Before that, Oxygen was very toxic to most living organisms, and the "Great Oxidation Event" resulted in a mass extinction [14]. But it is by no means obvious how likely it was for evolution to learn how to use oxygen to make ATP (the cell's energy currency). Is the fitness landscape full of possible adaptations that allows for the use of oxygen? or is it that perhaps the "fitness optimum" is atop a wide base where hill-climbing would get you there from many locations in the sequence space (however slowly)? Affirmation of either of these would suggest that these types of adaptations are probable. On the other hand, we often assume that a random sample of biological sequences is quite unlikely to contain sequences that exhibit a given function. In a famous experiment, Bartel and Szostak tried to provide some estimate of this challenge by screening a large pool of randomly synthesized RNA molecules, which were tested for their ability to ligate certain products[15] . In their experiment, the frequency of detecting a functional sequence was roughly 10^{-14}.

To summarize, there is value in building better estimates of Φ for any landscape, first because it helps us optimize on the landscape, and second, because being able to estimate the landscape overall can help us understand the outcome of stochastic processes within it better. These are distinct problems, as in the second case, we would like the error of the estimator to be low on average, whereas in the optimization case, we simply care about finding high-performing variants. Depending on which objective we prioritize, or the type of model we desire, we might sample sequences within a landscape differently [16].

We will call a sequence with a particular arrangement of alphabet a *variant*. Often, in biology, the process of exploring a landscape involves trying many variants in parallel, which constitutes a *population*. In natural (evo-

lutionary) settings, it is often the case that multiple copies of the same variant exist in the population. In synthetic assays, it is more desirable to test unique variants (duplicates are included for calibration). We also assume a fixed population size. In the approaches discussed here, we importantly assume that the fitness landscapes are *static*, that is they don't change over time or as the frequency of different variants in the population change (those are called "seascapes"). I.e. a population evolving over a landscape does not change the properties of the sequence-to-function map. Generally speaking, these are not true for natural organismal evolution, because the context changes the map, but hold for biological sequence design. The difference between $\mathcal{E}$ (or population) trying to optimize on static landscapes versus one that attempts to optimize on a dynamic one is like the difference between an indoor rock climber and a tennis player. In one case the challenge is somewhat fixed, while in the second it is opponent and tournament dependent.

2.2 Why is this problem hard?

The problem of optimizing sequences on fitness landscapes is hard because their map Φ often has complex, non-convex structure and landscapes are unimaginably large. Biological sequences often exhibit significant amount of *epistasis*, which refers to the influence of genetic context on the effects of a change within a sequence. In the sequence design setting, if the effects of swapping one letter for another in a position (also termed *locus*) in the sequence were independent of what happened elsewhere, we could feasibly measure the effects of changing a letter to another [2], for every position in the sequence (e.g. [17]) and then design the optimal sequence by picking the best letter at each position. Hill-climbing processes can optimize very easily in this setting as well [18]. While these types of landscapes may ap-

---

[2]Population geneticists call a particular pattern of letters within particular loci, an *allele*.

proximate the vicinity of a particular sequence well[17, 13, 7, 19, 11], single-peaked landscapes are thought to be rare, and there is evidence to suggests that the global structure of landscapes can be quite heterogeneous [13].

Epistatic effects come in multiple forms, *magnitude* epistasis refers to cases where one allele's presence changes the magnitude of the effect of another one (e.g. diminishing returns: if you have an apple, a second apple is less appealing to you). *Sign* epistasis refers to situations where an allele's presence changes the sign of the effect of another allele (e.g. you are very sleepy, you drink coffee, you feel better, you now drink an espresso, you feel worse because you are anxious now, but no worse than you were before you had coffee). In the extreme cases, *reciprocal sign* epistasis can lead to cases where two alleles that are good independently, end up being deleterious together (or vice versa, e.g. menthol and coke may taste fine by themselves, but having them together is extremely unpleasant). Due to epistatic effects, optimisation on biological landscapes involves dealing with many local optima.

Secondarily, sequence spaces scale exponentially in sequence length L. The total number of possible sequences of a certain length is $|\Sigma|^L$ where L is the length of the sequence, and many sequences of different lengths may perform the same functions, often resulting in a huge space of possible candidates that exhibit similar functionalities. Even to perturb a sequence with a few mutations, a combinatorial number of choices are possible, which means that even with a perfectly accurate model in hand, brute-force search is not feasible. Moreover, if the landscape is not dense in viable solutions, and exhibits sign epistasis, hill-climbing algorithms may be subject to exponential search times, even to find local optima [20, 21]. It is natural to ask then, how and if evolution has solved this problem. There is evidence that evolutionary search is not an exceptionally great learning algorithm [3],

---

[3]For some static fitness landscapes Valiant [22] showed that standard evolutionary pro-

suggesting that the solution space cannot be too difficult (or otherwise we would not see evolution succeed). The most promising biological inventions that seem to remedy search times on such landscapes are recombination and the so-called "regeneration" processes where you reset your walks in the sequence space through gene duplication (ensuring you don't break your current best answer too much, and cap your distance from a potential solution) [21, 24, 25].

2.3 How can machine learning help in solving this problem?

An exploration process on a fitness landscape can be described much like a Generative Adversarial Network (GAN) [26]: (a) A generator that makes a set of proposal sequences (b) A critique that evaluates the sequences produced by the generator (see Figure 1). Nature is the ultimate critique, but asking it questions is expensive. If we have a model that can propose functional sequences, then we can leave the job of the critique to nature. However, if our generator has a high rate of making poor choices (say it picks at random), then we would want to preface queries to nature by simulating the critique before committing to synthesizing new sequences. This involves building accurate models Φ' and using them as critiques. However, unlike GANs, error-propagation is not always possible throughout the entire chain, depending on our choice of generator and critique (you can't train nature's parameters).

All algorithms (natural or otherwise) that explore fitness landscapes require both of these components. The trick for machine learning algorithms is to replace nature's generator to make better than random guesses or simu-

cesses cannot find optimal solutions when other learning algorithms do. A recent study by Kaznatcheev proposes that frequency-dependent dynamics (which we do not consider here) can overcome that barrier [23].
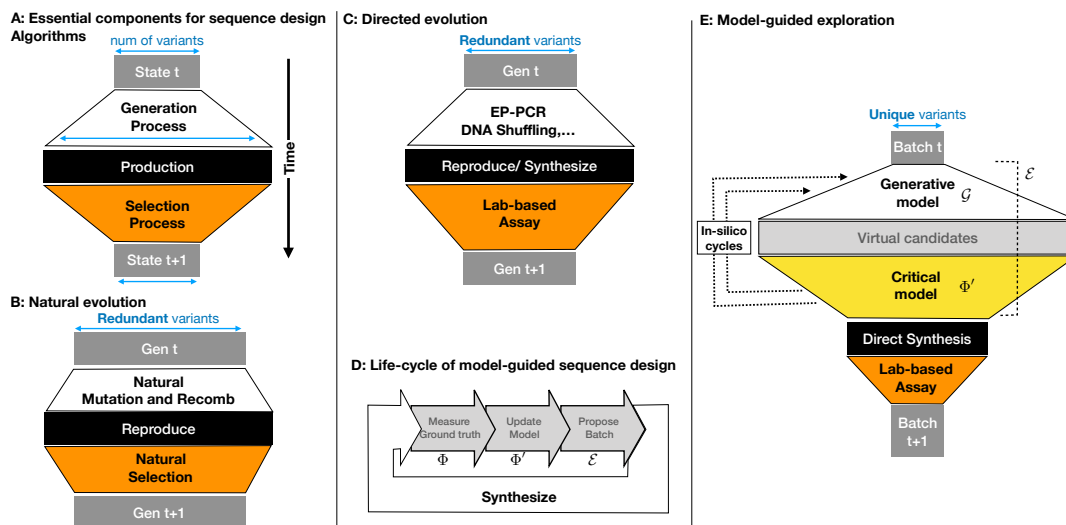
8

Figure 1: **A**: An exploration of the landscape has multiple stages. At each step, new candidates are proposed by a generative process, then they are produced and subsequently filtered by a selection process, which results in a new state for the population. **B**: For natural evolution, mutation and recombination are the generative process, physical reproduction synthesizes the variants, and natural selection filters them (as the critique). **C**: In directed evolution, generative processes are often lab-based methods of diversity production (e.g. error-prone PCR or in-vitro culturing) and then lab based assays are used to critique the variants that were produced by those processes. **D**: There are four steps in the life-cycle of model-guided sequence design. Candidates are **synthesized** and then **measured** experimentally to gather information about the landscape Φ. Given this information, models of landscape Φ' can be trained, and used by $\mathcal{E}$ to explore the sequence space, by proposing new samples for synthesis. **E**: In model-guided exploration, candidates may go through the virtual generation and critique (using Φ') steps multiple times before synthesis. This increases the chance of producing samples that will succeed in lab-based assays, reducing experimental costs and accelerating discovery.

late its critique to screen sequences before they are made (of course in both cases, they need synthetic biologists to make them, i.e. do the expensive step of asking nature).

In less metaphorical language, machine learning techniques that deal

with exploration and exploitation trade-offs can help propose better sequences. Machine learning methods that help us model Φ, either through black-box models or through better representations, can help us build better models of the landscape (oracles), thereby reducing the experimental costs.

3. How do we currently make models of fitness landscapes?

Fitness landscapes have been of interest to biologists for about a century[1]. There are two qualitatively distinct ways to approximate and study biological fitness landscapes: Empirical approaches based on experiments, and statistical approaches based on models which we will cover in detail. Empirical studies of fitness landscapes generally fall in two categories: First, studies in which samples are taken from an evolving population under natural or directed selection. These studies provide the benefit of producing diverse and large datasets that can be generated at once with little cost [27, 28]. They also are relatively faithful models of evolutionary process. However, by definition, most of these studies explore the "viable" regions of the landscape, leaving potentially interesting parts of the landscape untouched. It is hence difficult to know if a region of the landscape is non-viable or unexplored. Second, studies that sample variants using random or targeted mutagenesis [29, 2, 11, 13, 4, 30, 31]. In these studies one can synthesize a set of sequences and assay their performance in the laboratory. Advances in sequencing technology has allowed approaches such as error-prone PCR and DNA-reshuffling to generate sequences with random variation, where poor mutants can also be observed. Finally, direct synthesis approaches, the most recent development in the domain, allow for a large number of specifically designed edits to be synthesized and tested. The samples tested can

be precisely selected, hence they are of high informational value. However, mutational scanning is labor and resource intensive, and many orders-of-magnitude fewer samples are explored.

It is transparent that the two approaches are synergistic and can be combined. The first step has been to use available experimental data to infer consistent parameters for each class of models (described below). Encouragingly, simple models such as Additive, low-order polynomial models, and global epistasis, have been well-suited to describe the local properties of experimentally probed landscapes[7, 32, 33, 17]. Due to multiple limitations in how well models can capture complex landscapes for which long-range regularity is not well-understood [13, 34, 35], it is hard to predict the fitness of variants that are far from those already measured. A modern synthesis would likely involve mapping the insights gained from the empirical data back to refine the models that best describe these landscapes, hopefully in an interpretable, and generalizable manner. In this section, we will cover some classes of models that have provided useful insights over the years.

3.1   Mathematical and statistical models

For most of the time that the concept of fitness landscapes existed, measuring the effects of many mutations was hard. As a result, biologists have proposed multiple models of fitness landscapes that did not rely on empirical data, but aimed to emulate the properties of real landscapes.

The most fundamental but simple models are those in which the effects of mutations are independent, hence the collective effects of all mutations can be computed through an additive function. These landscape set the benchmark to compare all other landscapes with because at the core, additive models capture the effects of a letter (allele) in a particular position (locus) when considered independent of each other. Assuming we have a

11

sequence $x := \{\sigma_i \in \Sigma, \forall i | \sigma_1, \sigma_2, ..., \sigma_L\}$ of length L, the function (or fitness) $\Phi(x)$ can be defined as:

$$\Phi(x) = \sum_{i=1}^{L} \phi(\sigma_i)\omega_i \tag{1}$$

If all weights $\omega_i = 1$, and $\phi(\sigma_i)$ is the effect of a specific mutation at a specific location. This is the simple additive model[4]. However as described above, biological landscapes often deviate from additive (or linear) models, and a variety of approaches are designed to capture this deviation based on different perspectives.

In the most basic form (sometimes used as the "null" model) epistasis is completely unstructured (random). This is termed House of Cards (HoC) epistasis. Landscape models can be built from a mixture of HoC and additive effect to incorporate measurement noise or otherwise incompressible fitness effects[5].

A generalization of additive models is known as "Global epistasis" models where the additive baseline is transformed by a monotonic but non-linear function [7, 4, 36][6]. Despite their simplicity, and since additive landscapes can be easily inferred, these landscapes are somewhat successful in describing empirical mutational scan data sets [34, 2, 11, 35, 32], possibly because of the local regions that these scans tend to cover (there may be one dominant peak in the local neighborhood, as assumed in [19, 13]), which could be separated by large "valleys" [37].

---

[4]While generally used interchangeably, it's preferable to refer to the model as "linear" when $\omega_i \neq 1$.

[5]In a interesting recent study, Agarwala and Fisher investigate the dynamics of evolutionary walks on such landscapes that are locally additive (and correlated) but become less correlated as the distance to each peak is increased [19]

[6]A neural network based implementation can be found here: `https://mavenn.readthedocs.io/en/latest/`

$$\Phi(x) = \Gamma\Big(\sum_{i=1}^{L} \phi(\sigma_i)\omega_i\Big) + \eta \tag{2}$$

Where Γ is a non-linear function and η represents HoC epistasis.

A different group of elaborations on additive models aim to capture higher-order epsitatic interactions by assigning (or inferring) weights associated with interactions. In the most common form, they are written as coefficients of a regression.

$$\Phi(x) = \omega_0 + \sum_{i=1}^{L} \phi(\sigma_i)\omega_i + \sum_{i=1}^{L-1}\sum_{j=i+1}^{L} \phi(\sigma_i, \sigma_j)\omega_{ij} + ... \tag{3}$$

Where ω_i are weights on individual loci (positions), ω_{ij} are weights on pairwise interactions and so on. Inspired by Ising and Potts models, these can be used as generative descriptions of landscapes with certain amount of interaction among positions, or with some tricks, the parameters ω can be approximated efficiently [38, 39, 33, 40]. However, as efficient inference requires that the parameters are limited to a certain degree of interactions, often only linear and pairwise interactions are considered.

A more general approach introduced by Poelwijk and colleagues [30], emphasizes the role of background-averaging (observing mutations in multiple contexts, ideally all). This setup allows for the weights to be inferred across many degrees of interaction, decomposed through a Walsh-Hadamard transform (the discrete equivalent of a Fourier transform). The regression models can be imagined as the (local) Taylor expansion of the landscape centered at a particular reference, whereas Poelwijk et al's approach enables a global picture of the landscape akin to a Fourier decomposition [30, 41]. Provided such interactions are sparse across the sequence, they authors sug-

13

gest that compressive sensing can be used to infer these weights with a partial measurement of the landscape, allowing prediction.

Finally, a well-established set of landscape models is known as NK-landscapes [42], in which N is the length of the genome, and K is the number of positions in that genome that affect the fitness contribution of a single locus. The K parameter tunes the landscape for "ruggedness", where $K = 0$ denotes the additive model, and $K > 0$ describe landscapes with increasing number of local minima. While widely studied in the literature as they share qualitative features with empirical landscapes (and can be used to study *classes* of landscapes [20, 42]), they don't make good models of specific landscapes. Hence, they are more suitable for evaluating search processes across many different classes, however, it is unclear which ones would be biologically relevant, and even less clear if they are relevant for sequence design in a particular context.

3.2 Computational models

These models are simulators of the behavior of particular set of biological sequences, and they primarily focus on biophysical structure. The simulators are built on thermodynamical principles or other domain knowledge. These landscapes show considerable epistasis and complexity akin to those observed in natural landscapes (at least for specific regions of the landscape) [16, 34]. The drawback for these models is that they work only for a subset of problems where the biophysical first-principles are indeed sufficiently explanatory. An example of such problems would be thermostability of proteins, although not all thermostability problems are well-captured by these methods.

Two of the most famous examples of these simulators are those that simulate RNA sequence secondary structures (e.g. the Vienna package [43]), and

14

those that simulate protein stability and structure (e.g. Rosetta [44]). While an algorithm like Rosetta can be an excellent tool for design of de-novo proteins [45, 46], those that live in deep energetic valleys, it is less suitable at predicting the stability of natural sequences subject to perturbations, where the energy landscape is more subtle. As their strength, the type of bias that these models rely on are (by definition) close to well-established natural laws.

3.3   Machine learning models

A class of fitness landscape models, made available by the recent advances in deep mutational assays are fully data-dependent. These models, often black-box, are trained on data (both unsupervised and supervised), and are then used as an oracle for estimating fitness. Unsupervised models make use of the vast amount of unlabeled evolutionary sequence data that is available publicly to infer the fitness landscape (the Potts-model inspired work above also fall in this category) [47, 48, 49]. The assumption fundamental to these models is that similar evolutionary sequences are roughly capturing the same function. Supervised models depend on expensive assays to collect labels [50, 5, 4, 51, 52, 17], and often the assays are not designed for the models to optimally learn. In between these two limits there are semi-supervised approaches that try to take advantage of both sets of data [8, 9].

While it is possible to learn good models of the fitness landscape with these techniques, these black-box models are often not invertible and hence even with a perfect map of sequence to function, sequence design requires the ability to optimize. Additionally, the performance of these models are not uniform across the sequence space, and can vary drastically in different regions of the landscape. Therefore a desirable property of these models is to have them report their confidence in their estimates. These uncertainty estimates are inherent to models like Gaussian Processes [50], but can also

15

be achieved with neural networks by using techniques such as ensembling or dropout[53].

4. What does a good solution look like?

In this segment, we briefly cover the definitions and useful metrics by which sequence design algorithms may be evaluated. Often, sequence design is *online*, meaning that decisions have to be made as we collect information, and hence in retrospect, better decisions might have been possible. In this context, an "online algorithm" is an approach that provides certain guarantees for performance, whereas a " heuristic algorithm" is one that provides no such guarantees. Unless a particular regularity in the structure of the fitness landscape can be exploited, the majority of algorithms that we will describe provide no performance guarantees.

Experimental biology is hard, and often to get any measurement of the "ground-truth" function of a sequence takes significant time and effort. It is possible to do these measurements in batches of a certain size. Hence, if one has a budget to test a total of B sequences, and a maximum batch size of b, at each time step, the process gets to use the information gathered so far to propose sequences for the next batch. Practically, batch sizes can be of up to 10^6 for DNA synthesis, and each round of experiments can take a few months, and hence algorithms that perform better with fewer batches are more practical.

At their core, these algorithms need to balance the exploration-exploitation trade-off, based on the number of experiments that are made available to them (the "horizon"). Exploration can be seen as an information gathering act on the landscape, and exploitation is optimization of the desired outcomes based on that information.

4.1 Outcomes

Sequence design algorithms are considered useful if they can find a large number of distinct high performing sequences. We will break this heuristic down to two parts, optimality and diversity. These can be measured in multiple ways, and we discuss some simple metrics here.

For optimality, in most landscapes that these algorithms would be actually used for, critical information such as the best possible fitness y^* or the set of all local maxima $\mathcal{M}$ is unknown (otherwise, why use an algorithm to find them). Without loss of generality, we will assume that maximization is the objective, but it is noteworthy that while it is common to assume that the best sequence is the one with the highest value $\max(y)$, in reality the most desirable value of a trait is not necessarily the highest value. For instance, binding a particular target may be desirable, but binding it too strongly may be less desirable than binding it at a moderate level.

If the target values were known for a landscape, an algorithm can be considered efficient in its own right if it could find sequences $\mathbf{x}_i \in \mathcal{S}_\epsilon$ such that $||\Phi(\mathbf{x}_i) - \mathbf{y}^*|| < \epsilon$ with probability $p \geq 1 - \gamma$ for some acceptable tolerance $\epsilon, \gamma \geq 0$ and some bounded budget $B(L, \Sigma)$. Ideally, we would like $|S_\epsilon| >> 1$. However, we often do not have this information about the landscape in hand.

Without such information, one can assume the existence of a desired profile $\mathbf{y}^*$ and choose ϵ ad hoc (or perhaps anneal it) accordingly. Most approaches simply choose a single dimensional y and assume $y^* = \infty$, which translates to optimizing for $\max(y)$ for some bounded budget and batch size. An expansion would be to consider the cardinality $|\mathcal{S}_\tau|$, where $\mathcal{S}_\tau = \{\mathbf{x}_i \mid \Phi(\mathbf{x}_i) > y_\tau\}$ and $y_\tau \in \mathbb{R}$ is some minimum desired value. Notably, different algorithms may be better or worse in different budget and batch size regimes.

While finding a good solution is a primary goal, finding multiple solutions that are distinct is more desirable. This is because there are always reasons beyond the designer's control that can make a particular solution inadmissible. Having higher diversity hedges against this risk. Ideally, you don't get diversity by perturbing your best solution, but find very different sequences that are good at doing the same thing (find many maxima in multi-peaked fitness landscapes).

Similar to the optimality case, if we knew $\mathcal{M}$ we could use the number of found maxima $|\mathcal{M}'_{y_\tau}|$ where $\mathcal{M}'_{y_\tau} \in \mathcal{M}$ is the maxima found above fitness y_τ by the algorithm as a measure of diversity.

When we do not have access to the entire landscape, we can measure the diversity of sequences in $\mathcal{S}_\tau$. Unfortunately, summarizing the diversity of a set of sequences can be challenging when information about peaks is not available. High-dimensional clustering is has its own set of heuristic approaches, and we will not discuss them here. Simple metrics would include average pairwise edit distance, metric distance on embedded spaces, clustering, and site-specific entropy. None of these would uniquely describe the diversity of the set, and each are subject to their own drawbacks.

4.2 Properties

Apart from the outcomes of interest described above, there are also properties of the algorithm that would make it more practical or desirable. Depending on the type of problem, different properties could be prioritized.

4.2.1 Efficiency

We would like the algorithm to be efficient in two senses: (1) It should make fewer queries q_Φ to Φ (experimental samples) to achieve the same outcome and (2) We would like $q_{\Phi'}/q_\Phi$ to be small. This second property also relates

18

to scalability as it determines the computational resources required for the design of each batch.

4.2.2 Scalability

To be able to use our full experimental bandwidth our approach should be able to generate as many samples as our experimental batch of size B can accept. Furthermore, algorithms should scale as L grows. Note that it is possible to have algorithms that have low q_Φ and $q_{\Phi'}$, but are not scalable. For instance, they may be unable to produce enough diversity to fill a batch of 10^4 sequences due to mode collapse.

4.2.3 Consistency

If our models get better, our performance should also improve, i.e. $\mathcal{E}$ should make use of the information that is available to it.

4.2.4 Independence

Ideally, $\mathcal{E}$ should be independent of the model noise and bias. The algorithms shouldn't assume a particular type of noise or bias and we should be able to change the model or underlying landscape without making the algorithms useless [54]. I.e. if we switch the model class, we would still have a usable exploration algorithm.

4.2.5 Robustness

Closely related to independence, if the model is bad, (often because of misspecification [34]) the algorithm shouldn't fail completely. I.e. we would like the "worst-case" performance to be reasonably robust[54].

4.2.6 Adaptivity

In the computational sense, adaptivity denotes how many processes you can run in parallel. In experimental language, it captures the penalty for sampling N sequences at once (in one batch) as opposed to sampling them sequentially. The more adaptive a process is, the fewer samples need to taken serially, and hence, fewer batches are needed.

4.2.7 Reproducibility

We would like the algorithm to be reproducible. This means that it would be reasonably robust to hyper-parameters, easy to implement, and doesn't require extraordinary computational resources to perform well.

It is noteworthy that having an approach that excels in all of these criteria is often impossible. For instance, high adaptivity may come with the cost of lower independence overall (high adaptivity can be achieved by having an excellent model of the global landscape).

5. What approaches have been proposed to address this challenge?

5.1 Model-free approaches

5.1.1 Brute-Force search

Due to the combinatorial explosion of the landscape size as a function of the sequence length, the brute-force approach is feasible only on tiny landscapes or well-defined subsets of the fitness landscape, (e.g. all single mutants [17]). The generative process can simply enumerate all possible candidates, and records nature's response. This is the only model-free algorithm

with no stochastic element incorporated. All other well-known "deterministic" search schemes (E.g. BFS, DFS, Beam Search, ...) employed in this context make some decisions stochastically due to the high-branching factor or have to limit the search to a very small and biased sliver of the sequence space.

5.1.2 Random synthesis

On the other end of the spectrum, we can generate samples randomly. This is the equivalent of Bartel and Szostak experiments [15], where the generator is a random sequence generator, and the critique is nature. Given the very low success rate of the generator, this approach is impractical for optimization unless the problem is easy or the throughput is overwhelming.

5.1.3 Natural evolution

Nature's way of exploring fitness landscapes is by evolution. Evolution's generator is random perturbations on the population that has already been successful and survived so far, so while locally (almost) random, it contains a lot of information that was accumulated during the course of evolutionary history (This information is used in many unsupervised algorithms). Once sequences are generated through this process, they are directly passed to nature to decide what survives.

There two famous models of evolutionary processes, the Wright-Fisher process and the Moran process[55]. In the Wright-Fisher process, at each step N offspring are sampled by selecting a parent with probability proportional to their fitness, and replicating them (with some mutation rate μ). The entire new generation is made up of the offspring produced in the previous step, hence the generations are non-overlapping.

The Moran process [55] differs in that at each step only one member of

the current generation is sampled with probability proportional to their fitness and replicated, and one is removed at random. Hence the generations (every N replications) for the Moran process are overlapping. This results in a faster adaptation rate, as offspring in a given generation may descend from other high-fitness offspring within that same generation.

In both the Wright-Fisher and the Moran process, it is also possible to recombine with other offspring (or parent). This is in principle controlled by two parameters: (i) r which denotes the probability of a cross-over between two sequences already chosen to be recombining (ii) ρ which is the number of recombinations per offspring. It is common to set $\rho = 1$, which enforces sex to be between only two parents (this is not the case with some viral reproduction and DNA shuffling approaches). For comparison to batched experiments, the Wright-Fisher process appears a more suitable benchmark, however we expect natural populations to adapt at a rate closer to that of the Moran process (which is faster).

However, a more subtle aspect of fitness landscapes is the resolution at which two different traits would be considered different enough from evolution's perspective. Assuming at trait value y, we define the probability of sampling a mutant i for reproduction as $p_i = \frac{e^{\beta y_i}}{\sum_j e^{\beta y_j}}$ where β is the intensity of selection. Hence the relative fitness can be written as $e^{\beta(y_i - y_j)}$, and the minimum $\Delta_{ij} y = y_i - y_j$ for which selection overtakes genetic drift (effects from random sampling, independent of the phenotype) as the primary force is approximately $\frac{1}{\beta} \log(1 + 1/N)$. In other words, having a larger population (or stronger selection intensity) results in a landscape with higher resolution where a smaller difference in the trait value would be effectively "seen" by evolution.

Hence, evolutionary processes like the Wright-Fisher can be simulated with four parameters: population size N, mutation rate μ (which we define as the probability of an edit to a base or amino-acid), recombination rate r (im-

plicitly $\rho = 1$), and selection strength β. Evolutionary processes are often simulated in the limit where selection is strong and mutations occur infrequently (known as the strong selection, weak mutation limit SSWM), such that the population is monomorphic and updates one variant at a time (a variant appears and is selected to fixation, rather than having many competing variants in the population at once). This simplifies theoretical analysis significantly. However, this setting is unsuitable for measures like batch efficiency as you need many generations to achieve a high fitness, but is relatively sample efficient if you only care about unique samples tested.

Natural populations face the prospect of extinction. As a result, populations cannot tolerate very high mutation rates. For example small populations can quickly run out of good variants if the mutation rate is too high, resulting in further shrinkage of the population and exacerbation of the problem[56]. Even for large populations (e.g. viral populations), natural organisms show mutation rates below what is known as the *error threshold* defined as $\mu < \frac{1}{L}$, where L is the size of the genome [57]. But in our fixed population size models with no minimum fitness criteria, something will eventually survive, and hence it is noteworthy that optimal mutation rates in these conditions are possibly above the error threshold. Aside from these limitations, evolutionary processes in nature often benefit from achieving extremely large N, something that is far more limited in all the following strategies[7].

5.1.4 Directed (in lab) evolution

Classic directed evolution techniques take the principles of evolutionary search described above and apply it to problems of interest in lab. However, they also eliminate some of the constraints on natural evolution. They share the

---

[7]However, in nature, the census population size is different (larger) than the equivalent (effective) population size N that we use in the WF process, due to factors like non-random mating.

aspect that the generator is a set of random perturbations on an already functional population. However, in lab, we can increase the recombination frequency, i.e. $\rho >> 1$ and $\mu >> 1/L$ to generate farther diversity, as we get second attempts if a population of mutants is completely dead. On the critique side, they often bias and enhance the selective force β toward a particular phenotype of interest. This is a double-edged sword, on one hand this allows the power of evolution to be focused on a trait of interest, on the other hand it may result in deterioration of secondary traits that are also important and multi-objective directed evolution is challenging (see [58] for a modern framework towards this).

While these family of approaches have been extremely successful (and Francis Arnold won a Nobel prize for pioneering them [59]), it's been obvious for a while that at least on the choice of the what population to use as parents, augmenting them with a model (critique) can help with efficacy of the process [50, 60, 52].

5.2   Model-guided approaches

While the baselines introduced above are ones that have been the state of the art for many decades, with the advent of DNA synthesis technologies and rapid increase in the accessibility of machine learning, there is a lot of activity in the field of machine learning for protein engineering[48, 52]. The ambition is to surpass evolutionary search in how quickly we can find good solutions [10], and further decrease the time and effort required to do so (Note that all "model-free" approaches can be applied to optimise on Φ' instead of Φ).

5.2.1 In-silico evolutionary algorithms

Among the first studies that incorporated the entirety of the generate-model-explore framework was a series of papers by R. J. Fox and collegues [61, 60, 62] which first developed the concepts on NK-landscapes and then tried them on empirical ones. In these papers, the authors used in vitro and in silico **genetic algorithms** (random mutation and recombination) as their baseline generator in combination with Partial Least Squares (PLS) regression. In subsequent rounds used the weights of their model to inform the synthesis of mutants, hence, the model was employed both on the generator and critique side. Evolutionary algorithms are often a great choice to use for optimising on the landscape estimate. They can also be used in tandem with simple models (e.g. by structure [63]). The field of genetic and evolutionary algorithms is massive, and applied in many contexts [64, 65, 66]. We encourage the reader to explore the resources on quality-diversity algorithms linked in section 7. However, most of these efforts are done outside sequence design and have not been re-applied to biology itself, partially because they historically preceded the technical ability for direct synthesis. We have revisited this gap recently, and shown that simple and scalable evolutionary algorithms are competitive with the more modern approaches in terms of performance, robustness, and consistency [67]. These results suggest that evolutionary algorithms are still a benchmark to consider for model-guided sequence design.

5.2.2 Adaptations of classical explore-exploit algorithms

Sequential learning algorithms were developed in context where decisions have to be made based on partial observation of the data, and where choices may further inform the algorithm about the problem. In this online setting, the agent (explorer) needs to navigate the trade-off between exploiting the

current best known solutions, and exploring new solutions with the hope of improving on the best solution.

Among the most applied approaches adapted from this class of algorithms is the framework of Bayesian Optimisation (BO). BO algorithms are designed to optimize (possibly non-differentiable) black-box functions that are expensive to query (which is the setting we are in). Importantly, these algorithms make use of the uncertainty of model estimates to negotiate exploration vs. exploitation. As the algorithm starts out with no knowledge about the space we are trying to optimize on, a prior over the set of possible objective functions is assumed (hence Bayesian). The goal is then to update the prior belief with measurements to obtain a posterior distribution over the set of functions. The function that approximates the posterior distribution of y is known as the surrogate model (a more informative Φ'). Most BO approaches employ Gaussian Processes as their surrogate (see this tutorial: [68]).

An important aspect of optimization in this setting is to decide how to use uncertainty to prioritize measurements. In BO, this is termed as the "acquisition function". Some widely used acquisition functions are (i) Expected Improvement (EI) which picks the sample that has the highest expected improvement over the current best sample and (ii) Upper Confidence Bound (UCB) where the expected reward is augmented with an additional "optimism" term, for instance, by adding $\mu_r + k\sigma_r$ where μ_r is the mean posterior, and σ_r is the standard deviation of the posterior, and k is some scalar. Samples in UCB are then chosen as the one with the highest value of $\mu_r + k\sigma + r$.

In a pioneering study, Romero and colleagues [50] demonstrate the use of BO for protein engineering. Many productive efforts have followed since [69, 70, 71] and are well covered in this review [72]. While BO is a principled approach for optimization of black-box functions, it scales poorly in high-dimensional and high-throughput domains, such as those frequently

26

encountered in sequence design. Specifically how to define the domain of optimization, adapt to batch setting, represent sequence space, and choose good hyper-parameters can be challenging and differ significantly between experiments. Successful applications of BO to sequence design often require some domain art.

In the recent decade, another domain of study with applications to black-box optimization has flourished: Reinforcement Learning (RL). RL algorithms learn to perform tasks by experience, hence their success is often dependent on whether interactions with the environment are cheap or if there is a good simulator of the environment in which they can practice (e.g. chess). The "agent" that is guided by the algorithm interacts with the environment (or simulator), and observes rewards for taking different actions, over time, the agent learns to take actions with better reward. In our setting however, good simulators often don't exists, and sampling the environment directly is very expensive.

One exception is that of RNA secondary structure, for which good simulators exist (e.g. [43]). Eastman et al [73] take advantage of this to train reinforcement learning agents that are able to fold RNA sequences into particular secondary structures that are challenging to achieve. Most standard approaches to this problem include stochastic search (but generating random perturbations of current candidates), which results in inefficient sampling and reduced performance. Eastman et al, instead use graph-convolutional neural network to propose samples based on the networks knowledge of the simulated RNA-folding data, and train it with a reinforcement learning algorithm known as Asynchronous Advantage Actor-Critic algorithm (A3C).

Another promising approach is to build locally accurate simulators and use them to train an RL agent. Angemueller et al [74] train a policy network (a network that decides what mutations to make), by simulating the fitness landscapes through an ensemble of models. The models are trained on the

measured data so far, and those that achieve high R^2 in cross-validation are selected as "simulators". Agents are trained within this simulator up to the certain distance (determined by uncertainty in model estimates) from where data exists. Additionally, to increase the diversity of proposed sequences, they add a penalty for proposing samples that close to previously proposed samples (the closer they are, the higher the penalty).

The drawback of reinforcement learning approaches is the high computational (and domain expertise) cost of using these algorithms. In particular, PPO and TRPO algorithms are known to be highly implementation-sensitive [75].

5.2.3 Exploration on compressed representations of sequences

Biological sequences show a significant degree of regularities (e.g. motifs appear in sequences, which result in structural or functional properties). Hence, in principle the space of sequences that we are interested in can be compressed into a different representation, ideally continuous (an "embedding"), where we can use gradient-based optimisation techniques, or at least reduce the dimensionality of the data. Due to intense interest, these embedding methods are rapidly evolving, as they are also suitable for "semi-supervised" settings, where the embedding is learned using unlabeled natural sequences, and subsequently supervised models are trained on the smaller dataset (now embedded), with better performance.

Schemes to embed biological sequences fall into several general categories, notably VAEs [76, 77, 47, 78, 79] and invertible generative models (e.g. Flows or realNVPs) [80], as well as a plethora of increasingly promising models that have been adapted from natural language processing [81, 8, 49, 82, 83]. However, the exploration strategies implemented on these learned embeddings follow familiar Monte-Carlo, hill-climbing, or Bayesian Optimi-

28

sation schemes. I.e. these embeddings simplify the exploration problem it-self, rather than employing a complex optimisation on the original sequence space.

5.2.4   Regularized generative methods

The use of generative models to propose sequences with better properties is a natural route towards achieving the objective of producing diverse sequences that we did not observe within the training set. We highlight some of these approaches below.

Brookes and Listgarten [84] approach the exploration problem by carefully pairing a generative model with a regressor (oracle) that guides the generator towards it's own optima. Their first algorithm, titled Design by Adaptive Sampling (DbAS), assumes access to a static Φ'. Their algorithm works by training a generative model G_θ on a set of sequences x_0, and generating a set of proposal sequences $\hat{x} \sim G_\theta$. They then use Φ' to filter $\hat{x}$ for their high-performing sequences, retrain G_θ and redraw samples and iterate until convergence. This scheme is identical to the cross-entropy method with a VAE as the generative model (although they could use another generative model), an important optimization scheme [85]. Notably, the oracle is not updated during the process. I.e. out-of-the-box their process is described with two round of experiments in mind (a training set to make an oracle, and a resulting set proposed by the generative model), where they maximize the potential gains from their oracle, given what it already knows. While it is trivial to repeat the process for multiple rounds, the process can be improved by incorporating information about how many rounds it will be used for.

In follow-up work [6], Brookes et al, aim to improve the robustness of DbAS by introducing CbAS. This is meant to address the pitfall in which Φ' is bi-

ased, and gives poor estimates outside its training domain. The authors enforce a soft pessimism penalty for samples that are very distinct from those that the oracle could have possibly learned from. Specifically, they modify the DbAS paradigm such that as the generator updates its parameters $\theta_0 \rightarrow \theta_t$ while training on samples in the tail of the distribution, it discounts the weight of the samples x_i by $\frac{\mathbb{P}(x_i|G;\theta_0)}{\mathbb{P}(x_i|G;\theta_t)}$. In other words, if the generative model that was trained on the original data was more enthusiastic about a sample than the one that has updated according to the oracle's recommendations, that data point is up-weighted in the next training round (and *vice versa*).

Feedback-GANs (FBGAN) [86] approach sequence design using a GAN, which consisted of a generator G, and a discriminator D. In a standard GAN the generator's objective is to use a latent code z (often white noise) to produce samples x that are close to training data. The discriminator's objective is to discern if the x was generated by G or came from the ground truth distribution R (i.e. $D(x_i) = \mathbb{P}(x_i \in R)$). By jointly training G, D one can generate samples similar to those in the training set. Note however, that the discriminator does not necessarily provide an estimate of $\mathbf{y}$. To allow for optimizing $\mathbf{y}$, the authors pair the GAN with an oracle Φ'. At each epoch, multiple samples generated by G are passed through the oracle to acquire their label. Sequences with $\mathbf{y} > T$ (e.g. top quantile), are then presented to the discriminator for training. This ensures that over time, both D, G are biased towards proposing high-performing samples.

Killoran et al [87] pursue the same objective by regularizing an "inverted model", $\Phi'^{-1}(\mathbf{y}) = x$ through a generative adversarial network G. Activity maximization is a process where the input x is perturbed in small steps such that it improves the objective $\mathbf{y}$, i.e. $x^{(t+1)} = x^{(t)} + \alpha \nabla_x \mathbf{y}$ (α is the step size). For sequences, due to their non-continuous nature, this involves continuous relaxation schemes of the input through casting them as probabilities:

going from one-hot representations of sequences to position weight matrices (PWM). Activity maximization processes come with potential drawbacks however: (i) They can produce non-realistic sequences that do not look like those in the training data, (ii) they can be computationally expensive. The authors address the first problem by pairing the model with a generator (in their case a Wasserstein GAN) that is trained to produce samples similar to those that have the desired properties in the training set. The WGAN accepts a latent code z_i to generate sequence $x_i \sim G(z_i)$, and then pass them to $\Phi'(x_i) = \mathbf{y}$. The optimization recipe then becomes $z^{(t+1)} = z^{(t)} + \alpha \nabla_z \mathbf{y}$ where

$$\nabla_z \mathbf{y} = \sum_d \frac{\partial \mathbf{y}}{\partial x_d} \frac{\partial x_d}{\partial z}$$

where d represents the dimensions of x.

Deep Exploration Networks (DENs) [88], take a similar architecture as Killoran et al. but focus on optimizing the generator G only. They assume access to an oracle Φ' (pre-trained) that they do not optimize during training (i.e. the oracle training time is the offline overhead, which saves a lot of time compared to computing AM online). The main innovative aspect of this work is to force the generator to compete with itself to maintain diversity: a known pathology of GANs is to ignore the latent code and undergo "mode-collapse" (where the generator simply produces one type of sequence). To achieve this, the authors provide distinct latent codes to G, and penalize outputs $x_1 \sim G(z_1)$, $x_2 \sim G(z_2)$ that are similar. The cost function for this approach is defined as :

$$C = C_{objective}\Big(\Phi'(G(z_1))\Big) + C_{diversity}\Big(G(z_1), G(z_2)\Big)$$

which they seek to minimize by training G. This approach circumvents some of the drawbacks of AM, notably the computational cost of producing se-

quences. A simpler architecture with fast convergence has been proposed in follow up work [89].

5.3   Meta-algorithms

Of course, one is not restricted to use only one of the methods above to propose sequences. Angermueller et al. [90] propose a population-based method (P3BO) that ensembles exploration strategies together. The idea is very simple: at first many exploration strategies are given equal shares of the budget for each batch. Subsequently, given the performance of the sequences each proposed, the budget allocated to each method is updated, rewarding algorithms with better proposals. They show that this ensembling generally outperforms the baselines that only include single methods both in terms of diversity and optimality.

Another innovation that can be thought of as a meta-algorithm, applicable to many generative exploration strategies, is termed *auto-focusing* [91]. When a model is trained based on some collected data, and new samples are desired for the next batch that somehow optimize the properties of the input. Often, as discussed in this primer, the trained model is misspecified and starts to become inaccurate as the inputs deviate from those present in the training data. Search algorithms discussed above that query such models define a "trust region" (which can be soft or hard) and only accept designed inputs if they fall within that region. Auto-focusing is a recipe to instead retrain the predictive model using importance weighting, coupled with updates to the parameters of the search model (i.e. the generative explorer) in lock-step, to "focus" the model on the region of the search space that the explorer is visiting (i.e. improve the model's accuracy in that region), thereby improving the quality of the model (reducing its gap from the ground truth) in the region that the explorer is sampling.

5.4 Potential pitfalls

In this section, we list some considerations that are occasionally ignored in studies that have significance for evaluation and designing algorithms that can be used in practice.

5.4.1 Optimizing with pathological oracles

Ground truth oracles that are only proxies of ground truth (e.g. machine learning model of the landscape), can behave pathologically outside their training domain [92, 6, 74]. Algorithms that explore spaces built on these models can overfit in such landscapes: the biases introduced by these oracles can make an algorithm appear good in practice, when it is in fact a good optimizer for the model.

It is hence advisable to use ground-truth oracles that with consistent behavior across the entire exploration domain[67]. When not possible, a compromise would be to ensure that the algorithm's internal model is not from the same class as that of the surrogate oracle [68].

5.4.2 Allowing duplicate sequences for synthesis based approaches

When synthesis is the way in which sequences are generated, populations of sequences shouldn't contain duplicates, either within or between generations (except for experimental controls). Experimentalists don't need to remeasure sequences once they have a good measurement. This could artificially inflate diversity, and mean fitness measures. Natural and model-free directed evolution benchmarks are an exception.

5.4.3 Only testing on landscapes with gradients everywhere

A challenge that sequence design algorithms face in practical settings is that generating samples with aggressive changes may result in a batch of completely non-functional sequences. This would be costly and doesn't add much information to the models in order to improve the guesses for the next round. However, when test landscapes contain gradients everywhere, mistakes are not as costly, and algorithms can always re-initiate at a random place and optimize successfully. Testing on landscapes of this second kind only, can give results that are not generalizable to more realistic landscapes where vast areas of the sequence space contain no gradients (we term this as "swampland", but they've also mentioned as "holey" [50]).

5.4.4 Initializing optimization at random locations

This is related to the previous pitfall. Many realistic landscapes are swamplands, where random samples of sequences will not exhibit any function at all [15]. In such landscapes it is obvious to the designer that they need to start at sequences that already show some desired activity. However, this means that those sequences may already be at or near a peak. Algorithms that are very good hill climbers may not be great at escaping local peaks. Hence even when landscapes have gradient everywhere, it is advisable to test the algorithm on both high-performing starting samples as well as random (often poor) samples.

5.4.5 Miscalibration of selection pressure or population size when comparing with evolution

As discussed in section 5.1.3, the selective pressure and population size are highly important in the efficiency of evolutionary exploration. In particular, for any given population size, there fitness landscape is coarse-grained with

respect to the minimum difference in fitness Δy under which selection overtakes drift. In other words, evolution cannot "see" the difference between mutants that have $\Delta y << \frac{1}{\beta}\log(1 + 1/N)$. For this reason, one measure of efficiency or optimally, can be to state how fast a population reaches the same fitness that an evolving population with size N and selection intensity β would.

6. Conclusions

Sequence design with the aid of machine learning is a booming field that spans multiple disciplines. The topics covered in this primer are meant to educate and help initiate research from experts and beginners across these fields, and ease their introduction to this topic. This primer does not paint a full picture, and in particular, is light on its treatment of computational protein design. Additionally, given the rapid expansion of the field, we did not attempt to cover all recent progress in this domain. However, we hope this resource removes some of barriers for those entering the field. Fortunately, there is a lot still to be done, and we look forward to the upcoming progress.

7.   Further resources

- A list of quality-diversity algorithms:

  `https://quality-diversity.github.io/papers`

- A regularly updated repository of papers related to Model-guided protein design:

  `https://github.com/yangkky/Machine-learning-for-proteins.`

- A sandbox for evaluating sequence design algorithms:

  `https://github.com/samsinai/FLEXS.`

8.   Acknowledgements

References

[1] Sewall Wright. *The roles of mutation, inbreeding, crossbreeding, and selection in evolution*, volume 1. na, 1932.

[2] J Arjan Gm De Visser and Joachim Krug. Empirical fitness landscapes and the predictability of evolution. *Nature Reviews Genetics*, 15(7):480, 2014.

[3] Inna S Povolotskaya and Fyodor A Kondrashov. Sequence space and the ongoing expansion of the protein universe. *Nature*, 465(7300):922, 2010.

[4] Karen S Sarkisyan, Dmitry A Bolotin, Margarita V Meer, Dinara R Usmanova, Alexander S Mishin, George V Sharonov, Dmitry N Ivankov, Nina G Bozhanova, Mikhail S Baranov, Onuralp Soylemez, et al. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397, 2016.

[5] Surojit Biswas, Gleb Kuznetsov, Pierce J Ogden, Nicholas J Conway, Ryan P Adams, and George M Church. Toward machine-guided design of proteins. *bioRxiv*, page 337154, 2018.

[6] David H Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. *arXiv preprint arXiv:1901.10060*, 2019.

[7] Jakub Otwinowski, David Martin McCandlish, and Joshua Plotkin. Inferring the shape of global epistasis. *bioRxiv*, page 278630, 2018.

[8] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. Unified rational protein engineer-

ing with sequence-only deep representation learning. *bioRxiv*, page 589333, 2019.

[9] Surojit Biswas, Grigory Khimulya, Ethan C Alley, Kevin M Esvelt, and George M Church. Low-n protein engineering with data-efficient deep learning. *bioRxiv*, 2020.

[10] Eric D Kelsic and George M Church. Challenges and opportunities of machine-guided capsid engineering for gene therapy. *Cell and Gene Therapy Insights*, 2019.

[11] J Arjan GM de Visser, Santiago F Elena, Inês Fragata, and Sebastian Matuszewski. The utility of fitness landscapes and big data for predicting evolution, 2018.

[12] Sergey Kryazhimskiy, Daniel P Rice, Elizabeth R Jerison, and Michael M Desai. Global epistasis makes adaptation predictable despite sequence-level stochasticity. *Science*, 344(6191):1519–1522, 2014.

[13] Claudia Bank, Sebastian Matuszewski, Ryan T Hietpas, and Jeffrey D Jensen. On the (un) predictability of a large intragenic fitness landscape. *Proceedings of the National Academy of Sciences*, 113(49):14085–14090, 2016.

[14] Lynn Margulis and Dorion Sagan. *Microcosmos: Four billion years of microbial evolution*. Univ of California Press, 1997.

[15] D.P. Bartel and Jack Szostak. Bartel, d.p. and szostak, j.w. isolation of new ribozymes from a large pool of random sequences. science 261, 1411-1418. *Science (New York, N.Y.)*, 261:1411–8, 10 1993.

[16] Louis du Plessis, Gabriel E Leventhal, and Sebastian Bonhoeffer. How good are statistical models at approximating complex fitness landscapes? *Molecular biology and evolution*, 33(9):2454–2468, 2016.

[17] Pierce J Ogden, Eric D Kelsic, Sam Sinai, and George M Church. Comprehensive aav capsid fitness landscape reveals a viral gene and enables machine-guided design. *Science*, 366(6469):1139–1143, 2019.

[18] Herbert S Wilf and Warren J Ewens. Theres plenty of time for evolution. *Proceedings of the National Academy of Sciences*, 107(52):22454–22456, 2010.

[19] Atish Agarwala and Daniel S Fisher. Adaptive walks on high-dimensional fitness landscapes and seascapes with distance-dependent statistics. *Theoretical population biology*, 130:13–49, 2019.

[20] Artem Kaznatcheev. Computational complexity as an ultimate constraint on evolution. *Genetics*, 212(1):245–265, 2019.

[21] Krishnendu Chatterjee, Andreas Pavlogiannis, Ben Adlam, and Martin A Nowak. The time scale of evolutionary innovation. *PLoS computational biology*, 10(9), 2014.

[22] Leslie G Valiant. Evolvability. *Journal of the ACM (JACM)*, 56(1):1–21, 2009.

[23] Artem Kaznatcheev. Evolution is exponentially more powerful with frequency-dependent selection. *bioRxiv*, 2020.

[24] Michael J McDonald, Daniel P Rice, and Michael M Desai. Sex speeds adaptation by altering the dynamics of molecular evolution. *Nature*, 531(7593):233–236, 2016.

[25] Sam Sinai, Jason Olejarz, Iulia A Neagu, and Martin A Nowak. Primordial sex facilitates the emergence of evolution. *Journal of The Royal Society Interface*, 15(139):20180003, 2018.

[26] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[27] Benjamin H Good, Michael J McDonald, Jeffrey E Barrick, Richard E Lenski, and Michael M Desai. The dynamics of molecular evolution over 60,000 generations. *Nature*, 551(7678):45, 2017.

[28] Sandeep Venkataram, Barbara Dunn, Yuping Li, Atish Agarwala, Jessica Chang, Emily R Ebel, Kerry Geiler-Samerotte, Lucas Hérissant, Jamie R Blundell, Sasha F Levy, et al. Development of a comprehensive genotype-to-fitness map of adaptation-driving mutations in yeast. *Cell*, 166(6):1585–1596, 2016.

[29] Luis A Barrera, Anastasia Vedenko, Jesse V Kurland, Julia M Rogers, Stephen S Gisselbrecht, Elizabeth J Rossin, Jaie Woodard, Luca Mariani, Kian Hong Kock, Sachi Inukai, et al. Survey of variation in human transcription factors reveals prevalent dna binding changes. *Science*, 351(6280):1450–1454, 2016.

[30] Frank J Poelwijk, Michael Socolich, and Rama Ranganathan. Learning the pattern of epistasis linking genotype and phenotype in a protein. *Nature communications*, 10(1):1–11, 2019.

[31] José Aguilar-Rodríguez, Joshua L Payne, and Andreas Wagner. A thousand empirical adaptive landscapes and their navigability. *Nature ecology & evolution*, 1(2):1–9, 2017.

[32] Zachary R Sailer and Michael J Harms. Detecting high-order epistasis in nonlinear genotype-phenotype maps. *Genetics*, pages genetics–116, 2017.

[33] Thomas A Hopf, John B Ingraham, Frank J Poelwijk, Charlotta PI Schärfe, Michael Springer, Chris Sander, and Debora S Marks. Mutation effects predicted from sequence co-variation. *Nature biotechnology*, 35(2):128, 2017.

[34] Jakub Otwinowski and Joshua B Plotkin. Inferring fitness landscapes by regression produces biased estimates of epistasis. *Proceedings of the National Academy of Sciences*, page 201400849, 2014.

[35] Zachary R Sailer and Michael J Harms. Uninterpretable interactions: epistasis as uncertainty. *bioRxiv*, page 378489, 2018.

[36] Ammar Tareen, William Thornton Ireland, Anna Posfai, David Martin McCandlish, and Justin Block Kinney. Mave-nn: Quantitative modeling of genotype-phenotype maps as information bottlenecks. *BioRxiv*, 2020.

[37] Abe D Pressman, Ziwei Liu, Evan Janzen, Celia Blanco, Ulrich F Mueller, Gerald F Joyce, Robert Pascal, and Irene A Chen. Mapping a systematic ribozyme fitness landscape reveals a frustrated evolutionary network for self-aminoacylating rna. *Journal of the American Chemical Society*, 141(15):6213–6223, 2019.

[38] Debora S Marks, Lucy J Colwell, Robert Sheridan, Thomas A Hopf, Andrea Pagnani, Riccardo Zecchina, and Chris Sander. Protein 3d structure computed from evolutionary sequence variation. *PloS one*, 6(12), 2011.

[39] Magnus Ekeberg, Cecilia Lövkvist, Yueheng Lan, Martin Weigt, and Erik Aurell. Improved contact prediction in proteins: using pseudolikelihoods to infer potts models. *Physical Review E*, 87(1):012707, 2013.

[40] Jaclyn K Mann, John P Barton, Andrew L Ferguson, Saleha Omarjee, Bruce D Walker, Arup Chakraborty, and Thumbi Ndung'u. The fitness landscape of hiv-1 gag: advanced modeling approaches and validation of model predictions by in vitro testing. *PLoS computational biology*, 10(8), 2014.

[41] Edward D Weinberger. Fourier and taylor series on fitness landscapes. *Biological cybernetics*, 65(5):321–330, 1991.

[42] Stuart A Kauffman and Edward D Weinberger. The nk model of rugged fitness landscapes and its application to maturation of the immune response. *Journal of theoretical biology*, 141(2):211–245, 1989.

[43] Ronny Lorenz, Stephan H Bernhart, Christian Höner Zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F Stadler, and Ivo L Hofacker. Viennarna package 2.0. *Algorithms for molecular biology*, 6(1):26, 2011.

[44] Carol A Rohl, Charlie EM Strauss, Kira MS Misura, and David Baker. Protein structure prediction using rosetta. In *Methods in enzymology*, volume 383, pages 66–93. Elsevier, 2004.

[45] Andrew H. Ng, Taylor H. Nguyen, Mariana Gmez-Schiavon, Galen Dods, Robert A. Langan, Scott E. Boyken, Jennifer A. Samson, Lucas M. Waldburger, John E. Dueber, David Baker, and Hana El-Samad. Modular and tunable biological feedback control using a de novo protein switch. *Nature*, 2019.

[46] Hao Shen, Jorge A. Fallas, Eric Lynch, William Sheffler, Bradley Parry, Nicholas Jannetty, Justin Decarreau, Michael Wagenbach, Juan Jesus

Vicente, Jiajun Chen, Lei Wang, Quinton Dowling, Gustav Oberdorfer, Lance Stewart, Linda Wordeman, James De Yoreo, Christine Jacobs-Wagner, Justin Kollman, and David Baker. De novo design of self-assembling helical protein filaments. *Science*, 362(6415):705709, 2018.

[47] Adam J Riesselman, John B Ingraham, and Debora S Marks. Deep generative models of genetic variation capture mutation effects. *arXiv preprint arXiv:1712.06527*, 2017.

[48] Kevin K Yang, Zachary Wu, and Frances H Arnold. Machine learning in protein engineering. *arXiv preprint arXiv:1811.10775*, 2018.

[49] Alexander Rives, Siddharth Goyal, Joshua Meier, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv*, page 622803, 2019.

[50] Philip A Romero, Andreas Krause, and Frances H Arnold. Navigating the protein fitness landscape with gaussian processes. *Proceedings of the National Academy of Sciences*, 110(3):E193–E201, 2013.

[51] Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831, 2015.

[52] Zachary Wu, SB Jennifer Kan, Russell D Lewis, Bruce J Wittmann, and Frances H Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences*, 116(18):8852–8858, 2019.

[53] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep en-

sembles. In *Advances in neural information processing systems*, pages 6402–6413, 2017.

[54] Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ml predictions. In *Advances in Neural Information Processing Systems*, pages 9661–9670, 2018.

[55] Martin A Nowak. *Evolutionary dynamics.* Harvard University Press, 2006.

[56] Wilfried Gabriel, Michael Lynch, and Reinhard Bürger. Muller's ratchet and mutational meltdowns. *Evolution*, 47(6):1744–1757, 1993.

[57] Manfred Eigen. Selforganization of matter and the evolution of biological macromolecules. *Naturwissenschaften*, 58(10):465–523, 1971.

[58] Armita Nourmohammad and Ceyhun Eksin. Optimal evolutionary control for artificial selection on molecular phenotypes. *arXiv preprint arXiv:1912.13433*, 2019.

[59] Frances H Arnold. Design by directed evolution. *Accounts of chemical research*, 31(3):125–131, 1998.

[60] Richard Fox. Directed molecular evolution by machine learning and the influence of nonlinear interactions. *Journal of theoretical biology*, 234(2):187–199, 2005.

[61] Richard Fox, Ajoy Roy, Sridhar Govindarajan, Jeremy Minshull, Claes Gustafsson, Jennifer T Jones, and Robin Emig. Optimizing the search algorithm for protein engineering by directed evolution. *Protein engineering*, 16(8):589–597, 2003.

[62] Richard J Fox, S Christopher Davis, Emily C Mundorff, Lisa M Newman, Vesna Gavrilovic, Steven K Ma, Loleta M Chung, Charlene Ching,

Sarena Tam, Sheela Muley, et al. Improving catalytic function by prosar-driven enzyme evolution. *Nature biotechnology*, 25(3):338, 2007.

[63] Claire N Bedbrook, Austin J Rice, Kevin K Yang, Xiaozhe Ding, Siyuan Chen, Emily M LeProust, Viviana Gradinaru, and Frances H Arnold. Structure-guided schema recombination generates diverse chimeric channelrhodopsins. *Proceedings of the National Academy of Sciences*, 114(13):E2624–E2633, 2017.

[64] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[65] Kalyanmoy Deb. *Multi-objective optimization using evolutionary algorithms*, volume 16. John Wiley & Sons, 2001.

[66] Thomas Back. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996.

[67] Sam Sinai, Richard Wang, Alexander Whatley, Stewart Slocum, Elina Locane, and Eric Kelsic. Adalead: A simple and robust adaptive greedy search algorithm for sequence design. *arXiv preprint arXiv:2010.02141*, 2020.

[68] Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[69] Claire N Bedbrook, Kevin K Yang, Austin J Rice, Viviana Gradinaru, and Frances H Arnold. Machine learning to design integral membrane

channelrhodopsins for efficient eukaryotic expression and plasma membrane localization. *PLoS computational biology*, 13(10):e1005786, 2017.

[70] Javier Gonzalez, Joseph Longworth, David C James, and Neil D Lawrence. Bayesian optimization for synthetic gene design. *arXiv preprint arXiv:1505.01627*, 2015.

[71] Kevin K Yang, Yuxin Chen, Alycia Lee, and Yisong Yue. Batched stochastic bayesian optimization via combinatorial constraints design. *arXiv preprint arXiv:1904.08102*, 2019.

[72] Kevin K Yang, Zachary Wu, and Frances H Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694, 2019.

[73] Peter Eastman, Jade Shi, Bharath Ramsundar, and Vijay S Pande. Solving the rna design problem with reinforcement learning. *PLoS computational biology*, 14(6):e1006176, 2018.

[74] Christof Angermueller, David Dohan, David Belanger, Ramya Deshpande, Kevin Murphy, and Lucy Colwell. Model-based reinforcement learning for biological sequence design. In *International Conference on Learning Representations*, 2020.

[75] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep RL: A case study on PPO and TRPO. In *International Conference on Learning Representations*, 2019.

[76] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis

Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[77] Sam Sinai, Eric Kelsic, George M Church, and Martin A Nowak. Variational auto-encoding of protein sequences. *arXiv preprint arXiv:1712.03346*, 2017.

[78] Xinqiang Ding, Zhengting Zou, and Charles L Brooks III. Deciphering protein evolution and fitness landscapes with latent space models. *Nature Communications*, 10(1):1–13, 2019.

[79] Joe G Greener, Lewis Moffat, and David T Jones. Design of metalloproteins and novel protein folds using variational autoencoders. *Scientific reports*, 8(1):1–12, 2018.

[80] Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

[81] Kevin K Yang, Zachary Wu, Claire N Bedbrook, and Frances H Arnold. Learned protein embeddings for machine learning. *Bioinformatics*, 34(15):2642–2648, 2018.

[82] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. In *Advances in Neural Information Processing Systems*, pages 9686–9698, 2019.

[83] Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R Eguchi, Po-Ssu Huang, and Richard Socher.

47

Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*, 2020.

[84] David H Brookes and Jennifer Listgarten. Design by adaptive sampling. *arXiv preprint arXiv:1810.03714*, 2018.

[85] Reuven Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and computing in applied probability*, 1(2):127–190, 1999.

[86] Anvita Gupta and James Zou. Feedback GAN (FBGAN) for DNA: a novel feedback-loop architecture for optimizing protein functions. *arXiv preprint arXiv:1804.01694*, 2018.

[87] Nathan Killoran, Leo J Lee, Andrew Delong, David Duvenaud, and Brendan J Frey. Generating and designing DNA with deep generative models. *arXiv preprint arXiv:1712.06148*, 2017.

[88] Johannes Linder, Nicholas Bogard, Alexander B Rosenberg, and Georg Seelig. A generative neural network for maximizing fitness and diversity of synthetic dna and protein sequences. *Cell Systems*, 2020.

[89] Johannes Linder and Georg Seelig. Fast differentiable dna and protein sequence optimization for molecular design. *arXiv preprint arXiv:2005.11275*, 2020.

[90] Christof Angermueller, David Belanger, Andreea Gane, Zelda Mariet, David Dohan, Kevin Murphy, Lucy Colwell, and D Sculley. Population-based black-box optimization for biological sequence design. *arXiv preprint arXiv:2006.03227*, 2020.

[91] Clara Fannjiang and Jennifer Listgarten. Autofocused oracles for model-based design. *arXiv preprint arXiv:2006.08052*, 2020.

[92] Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. *arXiv preprint arXiv:1912.13464*, 2019.