Optimal Design of Stochastic DNA Synthesis Protocols based on Generative Sequence Models

Eli N. Weinstein
Harvard University
eweinstein@g.harvard.edu

Alan N. Amin
Harvard Medical School

Will Grathwohl
University of Toronto

Daniel Kassler
Harvard Medical School

Jean Disset
Harvard Medical School

Debora S. Marks
Harvard Medical School
debbie@hms.harvard.edu

Abstract

Generative probabilistic models of biological sequences have widespread existing and potential applications in analyzing, predicting and designing proteins, RNA and genomes. To test the predictions of such a model experimentally, the standard approach is to draw samples, and then synthesize each sample individually in the laboratory. However, often orders of magnitude more sequences can be experimentally assayed than can be affordably synthesized individually. In this article, we propose instead to use stochastic synthesis methods, such as mixed nucleotides or trimers. We describe a black-box algorithm for optimizing stochastic synthesis protocols to produce approximate samples from any target generative model. We establish theoretical bounds on the method's performance, and validate it in simulation using held-out sequence-to-function predictors trained on real experimental data. We show that using optimized stochastic synthesis protocols in place of individual synthesis can increase the number of hits in protein engineering efforts by orders of magnitude, e.g. from zero to a thousand.

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s).

1 INTRODUCTION

Large-scale nucleic acid sequencing and synthesis is integral to modern biology and biomedicine, from biotechnology to epidemiology to neuroscience to agriculture to evolutionary biology and beyond. Generative probabilistic modeling offers a rigorous framework for analyzing large scale sequencing data and forming predictions of new sequences that can be synthesized in the laboratory. Generative models have been used, for instance, to infer underlying structural and functional constraints on protein evolution, to predict pathogen sequences that may emerge in the future, and to predict novel enzyme sequences with desired functional properties (Marks et al., 2011; Hopf et al., 2017; Weinstein and Marks, 2021; Russ et al., 2020). In order to assay the properties of predicted sequences and discover novel functional sequences, samples from generative models must be synthesized in the laboratory at scale. Large libraries are particularly important for protein engineering applications, where they are screened for hits with rare properties, e.g. a particular catalytic or binding activity.

Unfortunately, synthesizing large numbers of samples from generative sequence models is challenging. The standard approach, which we refer to as "Monte Carlo (MC) synthesis", is to (1) sample from the model computationally, and then (2) synthesize each sample individually (Russ et al., 2020; Shin et al., 2021; Madani et al., 2021). In practice, however, MC synthesis is limited by cost: despite recent advances in synthesis technology, gene-length libraries typically do not exceed 10^4 unique sequences (Kosuri and Church, 2014). Far larger libraries, on the order of $10^6 - 10^{13}$, can be screened in many high-throughput assays. The set of likely sequences predicted by state-of-the-art generative models is often vastly larger still: a protein model

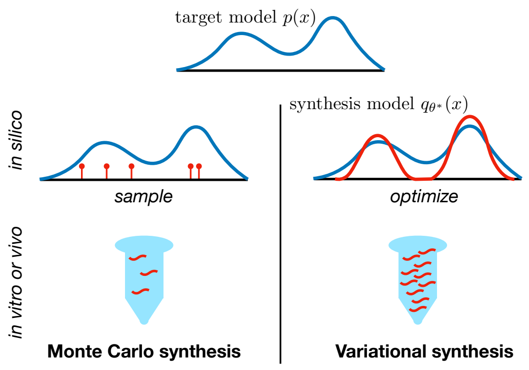

Figure 1: The standard synthesis approach for generative sequence models (Monte Carlo synthesis) is to sample sequences *in silico* and synthesize samples individually *in vitro*. The proposed approach (variational synthesis) is to optimize the experimental parameters of a stochastic synthesis protocol *in silico* and then run the protocol *in vitro* or *in vivo* to produce a larger number of samples.

with per-residue perplexity of 2 across sequences of length 100 predicts effectively $2^{100} \approx 10^{30}$ sequences. Thus MC synthesis often will come nowhere near comprehensive exploration of a model's predictions.

In principal, combinatorial and stochastic synthesis methods – such as error prone PCR and mixed nucleotides – offer an alternative approach capable of producing much larger numbers of unique sequences for the same cost. However, the sequences produced by these methods are random, and so it is unclear how to use stochastic synthesis to gain insight into the predictions of a given generative sequence model.

In this article, we describe an experimental design method – "variational synthesis" – that leverages stochastic DNA synthesis to overcome the limitations of MC synthesis. The basic idea is to optimize the parameters of the laboratory synthesis protocol to produce samples from a distribution close to the distribution of the target generative model. Variational synthesis is a rigorous approach to building ultra-large scale libraries based on generative sequence models, and can dramatically accelerate the discovery of novel functional sequences.

2 METHOD

We consider an arbitrary target generative model that describes a probability distribution $p(x)$ over sequences x. We are interested in assaying samples from the model experimentally. The standard method, MC synthesis, is to (1) draw samples $X_1, ..., X_{N_0} \sim p$ i.i.d. computationally and then (2) synthesize each sequence in the laboratory, deterministically. This approach is limited by the number of sequences N_0 that can be affordably synthesized deterministically, typically on the order of 10^4 or less for gene-length sequences.

As an alternative, we propose "variational synthesis" (Figure 1): (1) write down a probabilistic model $q_\theta(x)$ of sequences produced by a stochastic synthesis protocol with experimental parameters θ, (2) minimize a divergence between q_θ and p to find $q_{\theta^*} \approx p$ and (3) run the stochastic synthesis protocol in the laboratory, producing samples $X_1, ..., X_{N_1} \sim q_{\theta^*}$ i.i.d.. This approach is limited by the number of sequences N_1 that can be affordably screened, where in general N_1 can be orders of magnitude larger than N_0, e.g. $10^6 - 10^{11}$. The increase in samples comes at the cost of accuracy, since q_{θ^*} may not exactly match p.

2.1 Stochastic Synthesis Models

The first step of variational synthesis is to write down models q_θ of stochastic synthesis protocols. We focus on five key technologies: (1) enzymatic mutagenesis, e.g. error-prone PCR or Orthorep (Wilson and Keefe, 2001; Ravikumar et al., 2018), (2) mixed nucleotide synthesis, often referred to as "degenerate codon libraries" in the context of proteins (Pazdernik and Bowersox, 2016; Mena and Daugherty, 2005), (3) mixed trimer synthesis (Kayushin et al., 1996, 2000; McMahon et al., 2018), (4) combinatorial variant libraries (Twist Bioscience, 2020) and (5) combinatorial assembly (Gibson et al., 2009). We focus on models of protein sequences; models of DNA or RNA are simpler.

We describe stochastic synthesis models q_θ using a four-step generative process (Figure 2): (1) sample one of M "templates" from each of K "pools", (2) join the templates together, (3) sample codons independently at each position of the combined templates and (4) translate the DNA sequence into protein. For example, consider the protocol of combinatorial assembly plus error prone PCR: we start with a library of oligos, join (assemble) a random sample of oligos into a larger sequence, and then mutagenize the sequence. Abstractly, we refer to the distribution over codons obtained by mutagenizing a particular oligo as a "template". Techniques such as mixed nucleotides can produce alternative distributions over codons, described by different "templates". Mathematically, let $u_{kzj(b_1,b_2,b_3)}$ denote the probability of generating codon (b_1, b_2, b_3) at the jth position of template z in pool k. Let T be the translation matrix, defined as $T_{(b_1,b_2,b_3)d} = 1$ if the codon (b_1, b_2, b_3) codes for the amino acid d and $T_{(b_1,b_2,b_3)d} = 0$ otherwise. (For instance, $T_{(G,T,A)V} = 1$ since the codon GTA codes for

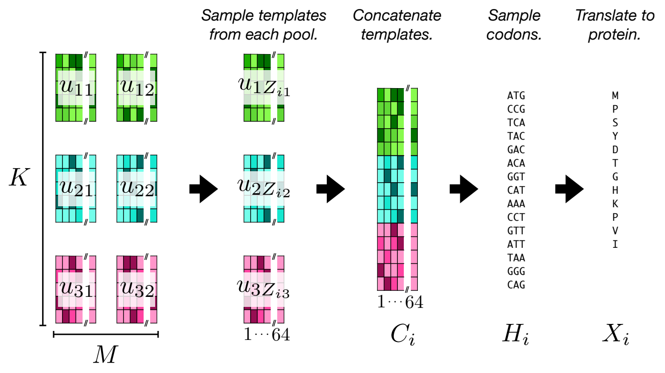

Figure 2: Overview of the synthesis model (Equation 1). From each of K pools we draw one of M templates, u_{kz}, according to the random vector Z_i. We concatenate the templates to form a matrix of codon probabilities C_i. Then codons are sampled at each position to form H_i, which is finally translated into a protein sequence X_i.

the amino acid V.) The complete model (Figure 2) is

$$\begin{aligned} Z_i &\sim p_w, \\ C_i &:= \text{concatenate}(u_{1Z_{i1}}, \ldots, u_{KZ_{iK}}), \\ H_i &\sim \text{Categorical}(C_i), \\ X_i &:= H_i \cdot T, \end{aligned} \tag{1}$$

where the "concatenate" operation stacks matrices vertically, and the categorical distribution produces one-hot encoded samples based on the probabilities in each row. Here, Z_i is the vector of templates used for sequence i, drawn from an underlying distribution p_w, while C_i is a matrix containing the codon probabilities for each site along sequence i and H_i is a one-hot encoding of the codons in sequence i (Table S1 provides a complete notation reference).

Different synthesis technologies impose different constraints on p_w, corresponding to different assembly methods, and different constraints on u, corresponding to different codon diversification methods. (The biochemical basis for these different mathematical constraints is described further in Section S2.) We consider two possible constraints on p_w:

1. Fixed assembly $Z_{i1} \sim \text{Categorical}(w)$ and $Z_{i2} := \ldots := Z_{iK} := Z_{i1}$. Here we assume that there are M templates in each pool, and that the choice of template from the first pool dictates the choice from all the others. The experimentalist can choose the probability vector $w \in \Delta_M$, where Δ_M denotes the $M-1$ simplex; chemically, w is controlled by the relative concentration of each template. In this case, the synthesis model (Equation 1) is a mixture model.

2. Combinatorial assembly: $Z_{ik} \sim \text{Categorical}(w_k)$ for all $k \in \{1, \ldots, K\}$. In this case each template from each pool is drawn independently. The experimentalist can choose the probability vectors $w_k \in \Delta_M$ for each pool.

We describe constraints on the codon probabilities of each template in terms of spaces $\mathcal{U}$, where the experimentalist can choose any $u_{kzj} \in \mathcal{U}$ for all k, z, j. We use $v \otimes v'$ to denote the outer product of two vectors v and v'. Overloading notation, for two sets of vectors S and S', we use $S \otimes S'$ to denote the set of outer products of their members, that is $S \otimes S' := \{v \otimes v' : v \in S \text{ and } v' \in S'\}$. We consider the following constraints:

1. Arbitrary codon mixtures: $\mathcal{U} = \Delta_{64}$. In this case, the experimentalist can choose any probability distribution over the 64 codons at each position in each template.[1] Combinatorial variant libraries have this constraint; it is the most flexible of the codon probability constraints we consider.

2. Finite codon mixtures: $\mathcal{U} = \{v_1, \ldots, v_A\}$ where $v_a \in \Delta_{64}$ for all a. In this case, the experimentalist must first fix a library of A different codon mixtures, and then, for each position in each template, choose one of these mixtures v_a to use. Mixed trimer synthesis protocols often have this constraint; in this case, v_a is determined by the relative concentration of each trimer in mixture a.

3. Finite nucleotide mixtures: $\mathcal{U} = \{v_1, \ldots, v_A\} \otimes \{v_1, \ldots, v_A\} \otimes \{v_1, \ldots, v_A\}$ where $v_a \in \Delta_4$ for all a. In this case, the experimentalist must first fix a library of A different *nucleotide* mixtures, and then, for each position in each codon in each template, choose one of these mixtures to use. Mixed nucleotide synthesis protocols often have this constraint; in this case, v_a is determined by the relative concentration of each nucleotide in mixture a

4. Enzymatic mutagenesis: $\mathcal{U} = \{S^\tau e_1, \ldots, S^\tau e_4\} \otimes \{S^\tau e_1, \ldots, S^\tau e_4\} \otimes \{S^\tau e_1, \ldots, S^\tau e_4\}$ where S is a substitution matrix, S^τ is a matrix exponential, and e_j is the length 4 vector of all zeros except a one at position j. The substitution matrix S is an intrinsic property of the chosen mutagenic enzyme (i.e. the particular error prone polymerase); in general, it has positive non-zero entries, linearly independent columns, and the sum of each column is 1. The number of rounds of mutagenesis $\tau \in \{1, 2, \ldots\}$ can be controlled experimentally.

Once an assembly technology (fixed or combinatorial) and codon diversification technology (arbitrary codon, finite codon, finite nucleotide or enzymatic) are chosen, the parameters θ of the synthesis model q_θ (Equa-

[1]We index the 64 codons either using either tuples $(A, A, A), \ldots, (T, T, T)$ or integers $1, \ldots, 64$, depending on convenience.

tion 1) that must be optimized consist of: w (the template probabilities), u (the codon probabilities), v (if we are using finite nucleotide or codon mixtures) and τ (if we are using enzymatic mutagenesis).

2.2 Black-Box Optimization

The second step of variational synthesis is to optimize the synthesis protocol, such that $q_{\theta^*} \approx p$. For some target/synthesis pairs – for instance, when the target is a regression model with a MuE output and fixed latent alignment (Weinstein and Marks, 2021), and the synthesis method uses fixed assembly and arbitrary codon mixtures – we can analytically and exactly match q_{θ^*} to p (Supplement S3.1). In most cases, however, an exact match between the target distribution and the synthesis distribution is impossible, and an analytic minimum intractable. We therefore propose an approximate optimization procedure. The primary desiderata are that it should be (1) black-box, in the sense that it can be applied to arbitrary target distributions p so long as p can be tractably sampled from, (2) scalable to large library sizes, since q_θ may for instance be a mixture model with 1000 or more components and (3) able to handle large numbers of discrete parameters, since $\mathcal{U}$ can be finite.

We propose to minimize the Kullback-Leibler (KL) divergence between the target model and the synthesis model, estimating $\theta^* := \text{argmin}_\theta \, \text{KL}(p \| q_\theta)$ by (1) drawing samples from the target model $X_1, \ldots, X_{\tilde{N}} \sim p$ i.i.d. and (2) maximizing the log likelihood of the samples under q_θ using a stochastic expectation-maximization (EM) algorithm (Cappé and Moulines, 2008). This approach only relies on samples from p, so can be applied whenever MC synthesis can be applied; in particular, it does not require access to likelihoods of p, allowing p to be an implicit model (e.g. a GAN). EM does not require access to derivatives of $q_\theta(x)$ with respect to θ, and can easily handle categorical parameters. Finally, since the stochastic EM algorithm relies only on minibatches of data, the method is highly scalable. Sections S3.2 and S3.3 detail the algorithm and provide advice on training, including the choice of $\tilde{N}$. Code is provided at `https://github.com/debbiemarkslab/variational-synthesis`.

Often the target p describes a distribution over variable-length sequences. One way to account for this, in the case of protein sequences, is to compute the likelihood of each sequence followed by a stop codon, treating the remainder of the DNA sequence as missing data when fitting q_θ (Supplement S3.4). Alternatively, a restriction site could be appended, and the remainder of the DNA sequence again treated as missing data; after synthesis, the sequences could be digested to the appropriate length. Our optimization procedure can thus be applied to p that produce variable-length sequences, so long as the length distribution is bounded.

3 RELATED WORK

Optimal design methods for stochastic synthesis have a long history, but existing techniques are in general non-probabilistic – they do not work with explicit target distributions p or synthesis distributions q_θ – and, practically, cannot be applied to produce samples from an arbitrary generative model p. Methods such as LibDesign (Mena and Daugherty, 2005) and SwiftLib (Jacobs et al., 2015) optimize degenerate codon libraries to match the per-position amino acid frequencies in a multiple sequence alignment, while limiting the total size of the library. SwiftLib has for instance been used to design massive libraries of mini-protein sensors and therapeutics (Chevalier et al., 2017; Klima et al., 2021). OCoM (Parker et al., 2011) applies similar ideas to handle pairwise correlations. The recent DeCoDe method (Shimko et al., 2020) designs degenerate codon libraries to produce as many members of a set of target sequences as possible, while limiting the total size of the library; it can be interpreted probabilistically as attempting to maximize the overlap in support between a synthesis distribution q_θ and a target distribution p, while regularizing the size of the support of q_θ (Section S4.1). Meanwhile, SCHEMA and RASPP (Voigt et al., 2002; Endelman et al., 2004) are used to optimize combinatorial assembly protocols based on protein structure, and have been applied to engineer new optogenetic tools (Bedbrook et al., 2017); when the target model p is a Potts model that accurately reflects protein structure, variational synthesis will prefer similar solutions (Section S4.2). Note that these existing non-probabilistic stochastic synthesis design tools are often used to construct libraries of diversified sequences in the context of directed evolution experiments, and we expect variational synthesis to also be applicable in the same context.

Batched stochastic Bayesian optimization (Yang et al., 2019) is comparable to variational synthesis in that it is a rigorous and probabilistic approach to stochastic synthesis optimization. Unlike variational synthesis, it is focused on optimizing a reward function, rather than drawing samples from a generative sequence model. It is also not black-box, relying on the particular structure of the reward function (a Gaussian process) and focusing on just one stochastic synthesis method.

Stochastic synthesis models related to those proposed in Section 2.1 have been used in the past for inference from observational data, rather than experimental design. For instance, Tomezsko et al. (2020) use a mixture model of sequences to infer RNA structural diversity from dimethyl sulfate mutational profiling data.

Variational synthesis is inspired by variational inference (VI) (Blei et al., 2017). Both minimize a divergence between a simple approximating distribution and a target distribution (a posterior in the case of VI). Both can take advantage of the expressiveness of mixture models to achieve close matches to the target distribution (Miller et al., 2016; Guo et al., 2016; Locatello et al., 2018). Both can be contrasted with older methods for exact sampling from a target distribution (Markov chain Monte Carlo in the case of VI, Monte Carlo synthesis in the case of variational synthesis); both trade accuracy for scale, enabling large numbers of approximate samples to be drawn (computationally in the case of VI, physically in the case of variational synthesis). Both can be black-box, enabling automatic sampling for a large class of target distributions (Ranganath et al., 2014; Kucukelbir et al., 2017).

4 THEORY

4.1 Approximation Error

In this section, we analyze the downstream consequences of using variational synthesis in place of MC synthesis. After synthesizing (approximate) samples from p, the sequences will be experimentally characterized using a high-throughput assay, described by a function f, which provides measurements $f(X_1), \ldots, f(X_N)$ of each synthesized sequence. The assay may measure binding strength, enzymatic activity, fluorescence, etc.. f is assumed to be unknown before performing the experiment. We consider two distinct goals. The first goal is to estimate the average value $\mathbb{E}_{X\sim p}[f(X)]$. For instance, we may want to estimate the average drug resistance of future pathogen sequences predicted by p. Second, we may be interested in discovering a large number of sequences with a desired property, i.e. we want to maximize $\sum_{i=1}^{N} f(X_i)$ where $f(x) = 1$ if the sequence has the property and $f(x) = 0$ otherwise. E.g. if we want to engineer a new plastic-degrading protein, we want to find as many sequences as possible with high degradation rates.

Estimating $\mathbb{E}_{X\sim p}[f(X)]$. MC synthesis and variational synthesis lead to two distinct estimators for $I := \mathbb{E}_{X\sim p}[f(X)]$, and in this section we compare their performance theoretically. In particular, the MC synthesis estimator is $\hat{I}^{(a)} := \frac{1}{N_0}\sum_{i=1}^{N_0} f(X_i)$ where $X_1, \ldots, X_{N_0} \sim p$, while the variational synthesis estimator is $\hat{I}^{(b)} := \frac{1}{N_1}\sum_{i=1}^{N_1} f(X_i)$ where $X_1, \ldots, X_{N_1} \sim q_{\theta^*}$. We have no *a priori* knowledge of f, so to compare estimators we evaluate worst-case performance over a family of functions $\mathcal{F}$. In practice, nearly all experimental assays have limited dynamic range; we therefore take $\mathcal{F}$ to be the set of bounded functions, $\mathcal{F} := \{f : \max_{x\in\mathcal{X}} |f(x)| \leq f_{\max}\}$, where $\mathcal{X}$ is the set of protein sequences of length less than or equal to L.

Proposition 4.1. *The worst-case mean absolute deviation of the exact synthesis estimator satisfies*

$$\frac{1}{f_{\max}} \sup_{f\in\mathcal{F}} \mathbb{E}[|\hat{I}^{(a)} - I|] \leq \frac{1}{\sqrt{N_0}}. \tag{2}$$

The worst-case mean absolute deviation of the stochastic synthesis estimator satisfies

$$\frac{1}{f_{\max}} \sup_{f\in\mathcal{F}} \mathbb{E}[|\hat{I}^{(b)} - I|] \leq \frac{1}{\sqrt{N_1}} + \sqrt{\frac{1}{2}\mathrm{KL}(p\|q_{\theta^*})}. \tag{3}$$

The proof, which can be found in Section S5.2, uses the integral probability metric representation of total variation along with Pinsker's inequality. This result describes a bias-variance tradeoff: using variational synthesis in place of MC synthesis leads to less variance (since $N_1 > N_0$) but introduces bias if q_{θ^*} does not exactly match p. Our optimization procedure (Section 2.2) minimizes bias by minimizing $\mathrm{KL}(p\|q_\theta)$.

If we have access to paired sequencing data, for instance if the hits of a high-throughput screen are deep-sequenced, we can remove the bias in the variational synthesis estimator via importance-weighting. We analyze this approach in Section S5.3.

Maximizing $\sum_{i=1}^{N} f(X_i)$. How many more hits can we expect to discover when using variational synthesis as opposed to MC synthesis? To address this question, we take $f : \mathcal{X} \mapsto \{0, 1\}$, and compare the total number of hits when using variational synthesis, $N_1\hat{I}^{(b)}$, to the number of hits when using MC synthesis, $N_0\hat{I}^{(a)}$.

Corollary 4.2. *The expected increase in hits when using variational instead of MC synthesis satisfies*

$$\begin{aligned}&\mathbb{E}[N_1\hat{I}^{(b)} - N_0\hat{I}^{(a)}] \geq \\ &\left(I - \sqrt{\frac{1}{2}\mathrm{KL}(p\|q_{\theta^*})}\right)N_1 - \sqrt{N_1} - IN_0 - \sqrt{N_0}.\end{aligned} \tag{4}$$

See Section S5.4 for a proof. In general N_1 is much larger than N_0, so the determining factor as to whether variational synthesis outperforms MC synthesis is whether q_{θ^*} is a sufficiently close approximation to p, i.e. whether $\sqrt{\frac{1}{2}\mathrm{KL}(p\|q_{\theta^*})} < I$. If so, the payoff from using variational synthesis can be substantial: to first order, the number of hits increases linearly with the number of sequences N_1. Our optimization procedure maximizes the lower bound on the number of hits by minimizing $\mathrm{KL}(p\|q_\theta)$.

4.2 Performance Limits

We have seen that the success of variational synthesis is determined by how closely q_θ can match the target p.

In this section, we analyze how closely the stochastic synthesis models described in Section 2.1 can match arbitrary target distributions p.

Limits on fixed assembly. We start by showing that synthesis protocols that use fixed assembly, and do not use enzymatic mutagenesis, can match any target distribution p arbitrarily well. We use $q_\theta(x|z)$ as shorthand for $q_\theta(x|Z_{i1} = z)$, the synthesis model distribution conditioned on the choice of template (mixture component). Let $\mathcal{P}(\mathcal{X})$ denote the set of probability distributions over $\mathcal{X}$. Let $\mathrm{supp}(q_\theta(x|z))$ denote the support of the distribution $q_\theta(x|z)$, i.e. the set of all $x \in \mathcal{X}$ such that $q_\theta(x|z) > 0$.

Proposition 4.3. *When using either* ***arbitrary codon mixtures***, ***finite codon mixtures*** *(with $A \geq 21$), or* ***finite nucleotide mixtures*** *(with $A \geq 4$): for any $p \in \mathcal{P}(\mathcal{X})$ and $\eta > 0$ there exists some M and θ such that (1) $\mathrm{KL}(p\|q_\theta) < \eta$ and (2) $\mathrm{supp}(q_\theta(x|z)) = \mathcal{X}$ for all $z \in \{1, \ldots, M\}$. When using* ***enzymatic mutagenesis****: there exists some $p \in \mathcal{P}(\mathcal{X})$ and $\eta > 0$ such that for all M and θ, we have $\mathrm{KL}(p\|q_\theta) > \eta$.*

See Section S5.5 for a proof. The result says that as long as we are not using enzymatic mutagenesis, the target distribution p can be arbitrarily well approximated without resorting to individual synthesis (that is, without setting $q_\theta(x|z)$ to be a delta function). Fundamentally, the problem with enzymatic mutagenesis is its discreteness: a sequence can be mutated at minimum once, so there is a minimum non-zero codon probability, given by the properties of the enzyme. This sets a limit on the "resolution" of p that can be matched by the synthesis procedure.[2]

Limits on combinatorial assembly. We next show that any synthesis protocols using combinatorial assembly cannot closely match arbitrary targets p even in the limit that the library size M goes to infinity. The result holds for any choice of $\mathcal{U}$.

Proposition 4.4. *When using* ***combinatorial assembly****, so long as $K > 1$, there exists $p \in \mathcal{P}(\mathcal{X})$ and $\eta > 0$ such that for all M and θ, we have $\mathrm{KL}(p\|q_\theta) > \eta$.*

See Section S5.6 for a proof. The key problem with combinatorial assembly is that it forces templates to be independent of one another; it therefore cannot match probability distributions p which have correlations between regions covered by each template.

[2]In practice, despite the mathematical idealization of our models, all synthesis technologies have a minimum non-zero codon probability, set by engineering constraints. The key question is really how low this number is comparatively.

5 RESULTS

5.1 Matching Evolutionary Enzyme Models

We next evaluated the ability of variational synthesis to produce approximate samples from target protein models trained on real data. As a first target, we chose a Potts model trained on dihydrofolate reductase (DHFR) sequences from across evolution; DHFR is an enzyme crucial for nucleic acid synthesis. Potts models of protein sequences have been studied extensively, and MC synthesis from Potts models can produce functional sequences (Russ et al., 2020). We optimized each of our proposed stochastic synthesis models, setting hyperparameters based on commercially-available technologies (Section S6.2). We compared our proposed variational synthesis approach to a baseline heuristic library diversification strategy of MC synthesis plus mutagenesis: (1) draw samples from p and then (2) apply five rounds of mutagenesis with ePCR (Section S6.3). To evaluate how well each synthesis model matched the target distribution we estimated its per residue perplexity (Section S6.4). However, perplexity only provides a measurement of the relative quality of different synthesis procedures, rather than an absolute measurement of whether they match the data distribution. We therefore applied a Bayesian two-sample test for biological sequences – the BEAR test (Amin et al., 2021) – to determine whether q_{θ^*} in fact matches p, based on 100,000 samples from each (Section S6.5).

All variational synthesis methods dramatically outperform the baseline (Figure 3A), and some are capable of matching the target p closely, passing the two-sample test (Figure 3B). Two key determinants of the performance of the stochastic synthesis model are (1) the expressivity of the codon diversification method – that is, the size of the set of allowed $\mathcal{U}$ – and (2) the number of templates M (Section S6.2). Performance in terms of perplexity shows an improvement with increasingly large $\mathcal{U}$ and increasing M. Note that due to current technology costs, when using codon mixtures, M must in general be small (e.g. ≤ 10) as compared to enzymatic mutagenesis or nucleotide mixtures (where M can be on the order of 1000). Nonetheless, using arbitrary codon mixtures with $M = 1$ templates outperforms the alternative technologies with $M = 1000$ templates.

The advantages of combinatorial assembly over fixed assembly vary depending on the codon diversification technology. Combinatorial assembly improves perplexity when using enzymatic mutagenesis, but has little effect when using arbitrary codon mixtures (Figure 3C and Figure S3), while introducing error in the covariance matrix of q_{θ^*} (Figure S4).

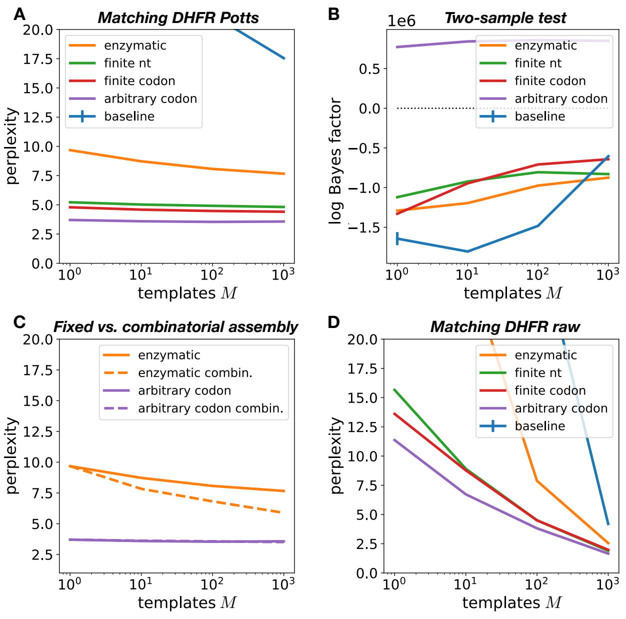

Figure 3: Perplexity (A) and two-sample test Bayes factor (B) of different codon diversification methods, with fixed assembly, applied to a target Potts DHFR model. Positive Bayes factors support the hypothesis that the synthesis and target distributions match. (C) Perplexity of combinatorial versus fixed assembly, applied to Potts DHFR model. (D) Perplexity of synthesis models with fixed assembly applied to unaligned DHFR sequences. Error estimates for each plot are described in detail in Section S6.7.

We next explored the application of variational synthesis to target distributions over variable-length sequences (the DHFR Potts model was trained on aligned sequences and generates fixed-length sequences). We optimized synthesis models directly on the same evolutionary data used to train the DHFR Potts model (with gaps removed); the target here is the true evolutionary data-generating process, and unknown (Section S6.1.1). Enzymatic mutagenesis with large M outperforms arbitrary codon mixtures with small M in this case (Figures 3D and S5). The best synthesis technology can thus depend on the target.

5.2 Synthesizing Fluorescent Proteins

Next we sought to determine if variational synthesis can increase the number of discoveries in downstream assays, as compared to MC synthesis. To simulate the results of realistic experimental assays, we used sequence-to-function predictors trained on large-scale experimental studies. We started with green fluorescent protein (GFP), predicting fluorescence using a transformer-based semi-supervised method trained on a GFP deep mutational scan dataset and evolutionary protein data (Sarkisyan et al., 2016; Rao et al., 2019). We classified as hits sequences with predicted fluorescence above the functionality threshold specified by Sarkisyan et al. (2016) (Section S6.6.1). To construct a target p, we trained an unsupervised sequence model – an ICA model with MuE output, proposed in Weinstein and Marks (2021) – on evolutionarily related GFP sequences, and then fixed the latent alignment variable of the MuE to generate sequences (Section S6.1.2). Using a fixed latent alignment ensures that the fluorescence predictor, which was only trained on fixed-length sequences, can be confidently applied. Note that the fluorescence predictor was not used to construct p itself, so we can fairly evaluate variational synthesis in the setting where the experimental results are not known ahead of time. In general, the fluorescence predictions are quite sensitive to the input sequence – a single amino acid change can abolish fluorescence – so generating new fluorescent sequences is nontrivial (Figure S6). Only 1.3% of sequences sampled from p are hits, with fluorescence above the threshold specified by Sarkisyan et al. (2016).

Stochastic synthesis models with arbitrary codon mixtures and fixed assembly have low perplexities, and can pass the two-sample test with large Bayes factors at $M \geq 10$; other methods struggle, including the baseline method (Figure 4AB). Samples from arbitrary codon models at $M = 10$ show average fluorescence similar to p (Figure S8), and the fraction of samples that are hits is only about half that of MC synthesis, 0.5% (Figure 4C). Meanwhile, alternative stochastic synthesis methods show hit rates below 0.05%.

Variational synthesis leads to a decrease in hit rate relative to MC synthesis, but this can be more than compensated for by the increase in the number of synthesized samples. If, for instance, $N_1 = 10^6$ sequences generated via variational synthesis are assayed, as opposed to $N_0 = 10^3$ sequences generated via MC synthesis, an estimated 3600 unique functional sequences will be discovered using variational synthesis as opposed to 10 for MC synthesis (Figure 4D; Section S6.6.4). Variational synthesis can thus provide orders-of-magnitude increases in the number of hits in protein engineering applications, with the number of hits increasing with larger values of N_1 and/or M.

5.3 Synthesizing Antigen-Binding Proteins

Next we sought to evaluate the advantages of variational synthesis over MC synthesis in an application area important for human health. Understanding T cell receptor (TCR) sequences and their binding properties is crucial for understanding the immune response to infection or cancer, and engineering new

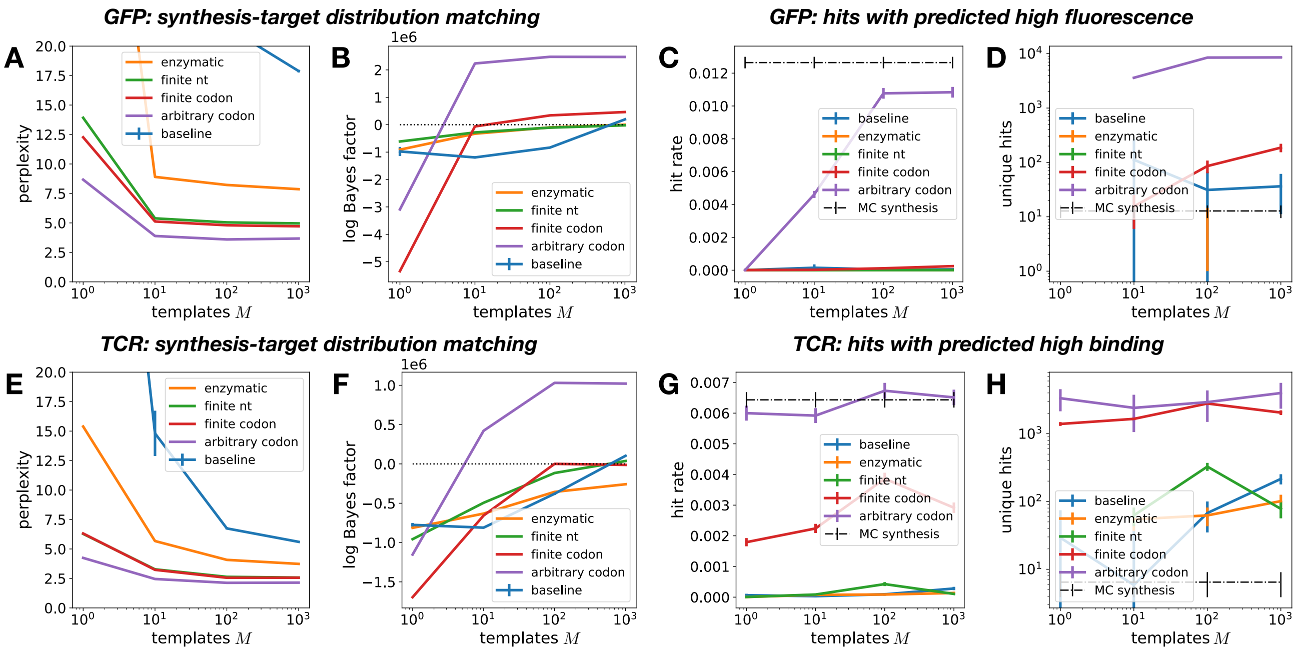

Figure 4: Perplexity (A) and two-sample test Bayes factor (B) for different synthesis methods applied to a target GFP model. (C) Hit rate for discovering functional sequences. (D) Expected number of unique hits in a $N_1 = 10^6$ library for variational synthesis, as compared to MC synthesis with a $N_0 = 10^3$ library (Section S6.6.4). (E-H) Same as (A-D) for a target TCR model. Error estimates for each plot are described in detail in Section S6.7.

TCRs with desired binding properties is crucial for immunotherapies (June et al., 2018). We trained a model of TCR sequences from a healthy donor – an ICA model with MuE output – and fixed the latent alignment variable in the MuE to define p (Section S6.1.3). As a held-out sequence-to-function predictor, we used Tcellmatch (Fischer et al., 2020) to predict binding to an influenza epitope (Section S6.6.2). The predictor is highly sensitive to the input sequence – a single amino acid change can abolish binding – making this a challenging problem for variational synthesis (Figure S10). Only 0.6% of samples from the target p are hits.

Synthesis models with arbitrary codon mixtures and fixed assembly achieve low perplexities and can pass the two-sample test with large Bayes factors (Figure 4EF). Variational synthesis with this model achieves hit rates similar to MC synthesis (Figure 4G). MC synthesis with $N_0 = 10^3$ generates just 6 hits on average across independent libraries; given stochasticity, it is not unlikely to see no hits at all in a given library. Variational synthesis with $N_1 = 10^6$ and $M = 10$ generates an expected 2400 unique hits (Figure 4H). Similar results hold for additional epitopes, from other viruses (Section S6.8.3). These results suggest that variational synthesis can dramatically accelerate the discovery of new TCRs that bind specific antigens, relying only on unsupervised sequence models and not large-scale supervised sequence-to-function training data.

Close matches between q_{θ^*} and p turn out to be unnecessary for reaching high hit rates in this example. When using arbitrary codon mixtures or finite codon mixtures with $M = 1$, or even using finite nucleotide mixtures with $M = 100$, the two-sample test detects significant differences between q_{θ^*} and p (Figure 4F), but nonetheless variational synthesis achieves substantially more hits than MC synthesis (Figure 4H).

6 DISCUSSION

Variational synthesis trades accuracy for scale, producing large numbers of approximate samples from a target model rather than small numbers of exact samples, as in MC synthesis. When accuracy is high enough – when q_{θ^*} is sufficiently close to p – the payoff can be enormous, as the number of hits increases linearly with the number of assayed sequences N_1. Given that many high-throughput screens can reach 10^{10} sequences or more, while individual gene synthesis rarely goes beyond $N_0 = 10^4$, using variational synthesis may make the difference between zero hits and a million.

We have shown through detailed simulations that such large payoffs are plausible for real, therapeutically important protein design targets, using commercially available stochastic synthesis technology. Going forward, implementing variational synthesis experimentally is thus a matter of ordering and assaying commercially-made libraries based on q_{θ^*}.

The key limitations of our variational synthesis methods – and opportunities for future work – stem from the challenges of matching synthesis and target distributions. First, our synthesis models (Section 2.1) are idealizations based on manufacturers' descriptions of the distribution of sequences their methods produce, but do not take into account possible errors, biases or limitations in the real procedure (Section S2). Developing more accurate q_θ models, based on e.g. deep sequencing data, may be an important area for future work. Second, our methods for judging whether q_{θ^*} is sufficiently close to p are limited. Empirically, while the BEAR two-sample test appears to be excellent at distinguishing among good and bad fixed assembly models in the examples we studied, it struggles to detect the errors caused by combinatorial assembly, even when they are large enough to abolish function (Figure S9). Theoretically, tighter bounds than that in Proposition 4.1 can be proved with total variation or Wasserstein distance in place of KL, but optimizing these alternative divergences directly is a challenge (Section S5.2). For sequence-to-function predictors to be more reliable in evaluating variational synthesis methods, they must be robust to covariate shift, since switching from p to q_{θ^*} is, precisely, a covariate shift. Third, while our black-box optimization method allows for arbitrary target distributions p, it may be more effective in many cases to work with p for which an exactly matching q_{θ^*} can be found analytically (Section S3.1). Recent progress on mixture models as a competitor to deep generative neural network models make this approach especially promising (Richardson and Weiss, 2018).

Variational synthesis changes the calculus of what makes a successful generative sequence model and what makes a successful synthesis technology. If just 1% of the sequences sampled from an initial model A were functional, and 50% of sequences sampled from a proposed model B were functional, model B would be considered a major advance; however, if we could accurately match a stochastic synthesis protocol to model A and not to model B, then model A could easily lead to orders of magnitude more hits in practice. Meanwhile, the traditional goal of the DNA synthesis community has been large-scale individual synthesis. From a probabilistic perspective, however, it hardly makes sense to focus exclusively on methods to sample from mixtures of point masses. The recent development of methods to synthesize samples from much more flexible mixture models represents a major advance outside the traditional paradigm.

Variational synthesis bridges the gap between generative sequence models and stochastic synthesis technologies, providing a rigorous approach to experimental design. We are optimistic that it will help translate powerful new generative sequence models into laboratory discoveries.

Acknowledgments

We thank Elizabeth Wood, Steven Weber, George Church, Tamara Broderick, Ed Boyden, Chris Sander, members of the Marks Lab and the anonymous reviewers for feedback and discussion. E.N.W. is supported by a Fannie and John Hertz Foundation Fellowship. D.S.M. is supported by the Chan Zuckerberg Initiative.

References

10x Genomics (2019). A new way of exploring immunity - linking highly multiplexed antigen recognition to immune repertoire and phenotype.

Alexandrov, L. B. and Stratton, M. R. (2014). Mutational signatures: the patterns of somatic mutations hidden in cancer genomes. *Curr. Opin. Genet. Dev.*, 24:52–60.

Amin, A. N., Weinstein, E. N., and Marks, D. S. (2021). A generative nonparametric bayesian model for whole genomes.

Bedbrook, C. N., Rice, A. J., Yang, K. K., Ding, X., Chen, S., LeProust, E. M., Gradinaru, V., and Arnold, F. H. (2017). Structure-guided SCHEMA recombination generates diverse chimeric channelrhodopsins. *Proc. Natl. Acad. Sci. U. S. A.*, 114(13):E2624–E2633.

Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P., Horsfall, P., and Goodman, N. D. (2019). Pyro: Deep universal probabilistic programming. *J. Mach. Learn. Res.*, 20(28):1–6.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *J. Am. Stat. Assoc.*, 112(518):859–877.

Cappé, O. and Moulines, E. (2008). On-line expectation–maximization algorithm for latent data models. *Journal of the Royal Statistical Society B*, 71:593–613.

Chevalier, A., Silva, D.-A., Rocklin, G. J., Hicks, D. R., Vergara, R., Murapa, P., Bernard, S. M., Zhang, L., Lam, K.-H., Yao, G., Bahl, C. D., Miyashita, S.-I., Goreshnik, I., Fuller, J. T., Koday, M. T., Jenkins, C. M., Colvin, T., Carter, L., Bohn, A., Bryan, C. M., Fernández-Velasco, D. A., Stewart, L., Dong, M., Huang, X., Jin, R., Wilson, I. A., Fuller, D. H., and Baker, D. (2017). Massively parallel de novo protein design for targeted therapeutics. *Nature*, 550(7674):74–79.

Dieng, A. B., Tran, D., Ranganath, R., Paisley, J., and Blei, D. (2017). Variational inference via chi upper bound minimization. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 2732–2741. Curran Associates, Inc.

Dragomir, S. S. (1999). Upper and lower bounds for Csiszar f-divergence in terms of the Kullback-Leibler distance and applications.

Dudley, R. M. (2002). *Real Analysis and Probability.* Cambridge University Press.

Eddy, S. R. (2011). Accelerated profile HMM searches. *PLoS Comput. Biol.*, 7(10):e1002195.

Efron, B. and Stein, C. (1981). The jackknife estimate of variance. *Ann. Stat.*, 9(3):586–596.

El Karoui, M., Hoyos-Flight, M., and Fletcher, L. (2019). Future trends in synthetic biology - a report. *Front Bioeng Biotechnol*, 7:175.

Endelman, J. B., Silberg, J. J., Wang, Z.-G., and Arnold, F. H. (2004). Site-directed protein recombination as a shortest-path problem. *Protein Eng. Des. Sel.*, 17(7):589–594.

Fischer, D. S., Wu, Y., Schubert, B., and Theis, F. J. (2020). Predicting antigen specificity of single T cells based on TCR CDR3 regions. *Mol. Syst. Biol.*, 16(8):e9416.

Gibson, D. G., Young, L., Chuang, R.-Y., Venter, J. C., Hutchison, 3rd, C. A., and Smith, H. O. (2009). Enzymatic assembly of DNA molecules up to several hundred kilobases. *Nat. Methods*, 6(5):343–345.

Guo, F., Wang, X., Fan, K., Broderick, T., and Dunson, D. B. (2016). Boosting variational inference.

Hopf, T. A., Ingraham, J. B., Poelwijk, F. J., Schärfe, C. P. I., Springer, M., Sander, C., and Marks, D. S. (2017). Mutation effects predicted from sequence co-variation. *Nat. Biotechnol.*, 35(2):128–135.

Jacobs, T. M., Yumerefendi, H., Kuhlman, B., and Leaver-Fay, A. (2015). SwiftLib: rapid degenerate-codon-library optimization through dynamic programming. *Nucleic Acids Res.*, 43(5):e34.

June, C. H., O'Connor, R. S., Kawalekar, O. U., Ghassemi, S., and Milone, M. C. (2018). CAR T cell immunotherapy for human cancer. *Science*, 359(6382):1361–1365.

Kayushin, A., Korosteleva, M., and Miroshnikov, A. (2000). Large-scale solid-phase preparation of 3'-unprotected trinucleotide phosphotriesters–precursors for synthesis of trinucleotide phosphoramidites. *Nucleosides Nucleotides Nucleic Acids*, 19(10-12):1967–1976.

Kayushin, A. L., Korosteleva, M. D., Miroshnikov, A. I., Kosch, W., Zubov, D., and Piel, N. (1996). A convenient approach to the synthesis of trinucleotide phosphoramidites—synthons for the generation of oligonucleotide/peptide libraries. *Nucleic Acids Res.*, 24(1):9–1996.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *ICLR*.

Klima, J. C., Doyle, L. A., Lee, J. D., Rappleye, M., Gagnon, L. A., Lee, M. Y., Barros, E. P., Vorobieva, A. A., Dou, J., Bremner, S., Quon, J. S., Chow, C. M., Carter, L., Mack, D. L., Amaro, R. E., Vaughan, J. C., Berndt, A., Stoddard, B. L., and Baker, D. (2021). Incorporation of sensing modalities into de novo designed fluorescence-activating proteins. *Nat. Commun.*, 12(1):856.

Kosuri, S. and Church, G. M. (2014). Large-scale de novo DNA synthesis: technologies and applications. *Nat. Methods*, 11(5):499–507.

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., and Blei, D. M. (2017). Automatic differentiation variational inference. *J. Mach. Learn. Res.*, 18:1–45.

Locatello, F., Khanna, R., Ghosh, J., and Ratsch, G. (2018). Boosting variational inference: an optimization perspective. In Storkey, A. and Perez-Cruz, F., editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 464–472. PMLR.

Madani, A., Krause, B., Greene, E. R., Subramanian, S., Mohr, B. P., Holton, J. M., Olmos, J. L., Xiong, C., Sun, Z. Z., Socher, R., Fraser, J. S., and Naik, N. (2021). Deep neural language modeling enables functional protein generation across families.

Marks, D. S., Colwell, L. J., Sheridan, R., Hopf, T. A., Pagnani, A., Zecchina, R., and Sander, C. (2011). Protein 3D structure computed from evolutionary sequence variation. *PLoS One*, 6(12):e28766.

McMahon, C., Baier, A. S., Pascolutti, R., Wegrecki, M., Zheng, S., Ong, J. X., Erlandson, S. C., Hilger, D., Rasmussen, S. G. F., Ring, A. M., Manglik, A., and Kruse, A. C. (2018). Yeast surface display platform for rapid discovery of conformationally selective nanobodies. *Nat. Struct. Mol. Biol.*, 25(3):289–296.

Mena, M. A. and Daugherty, P. S. (2005). Automated design of degenerate codon libraries. *Protein Eng. Des. Sel.*, 18(12):559–561.

Miller, A. C., Foti, N., and Adams, R. P. (2016). Variational boosting: Iteratively refining posterior approximations.

Moore, G. L. and Maranas, C. D. (2000). Modeling DNA mutation and recombination for directed evolution experiments. *J. Theor. Biol.*, 205(3):483–503.

National Research Council, Policy and Global Affairs, Committee on Science, Technology, and Law, and Committee on a New Government-University Partnership for Science and Security (2007). *Science and Security in a Post 9/11 World: A Report Based on Regional Discussions Between the Science and Security Communities.* National Academies Press.

Orlitsky, A., Suresh, A. T., and Wu, Y. (2016). Optimal prediction of the number of unseen species. *Proc. Natl. Acad. Sci. U. S. A.*, 113(47):13283–13288.

Parker, A. S., Griswold, K. E., and Bailey-Kellogg, C. (2011). Optimization of combinatorial mutagenesis. *J. Comput. Biol.*, 18(11):1743–1756.

Pazdernik, N. and Bowersox, A. (2016). Need a library of related DNA or RNA oligo sequences? `https://www.idtdna.com/pages/education/decoded/article/need-a-library-of-related-dna-or-rna-oligo-sequences`. Accessed: 2020-8-25.

Plesa, C., Sidore, A. M., Lubock, N. B., Zhang, D., and Kosuri, S. (2018). Multiplexed gene synthesis in emulsions for exploring protein functional landscapes. *Science*, 359(6373):343–347.

Potter, S. C., Luciani, A., Eddy, S. R., Park, Y., Lopez, R., and Finn, R. D. (2018). HMMER web server: 2018 update. *Nucleic Acids Res.*, 46(W1):W200–W204.

Pritchard, L., Corne, D., Kell, D., Rowland, J., and Winson, M. (2005). A general model of error-prone PCR. *J. Theor. Biol.*, 234(4):497–509.

Ramien, C., Yusko, E. C., Engler, J. B., Gamradt, S., Patas, K., Schweingruber, N., Willing, A., Rosenkranz, S. C., Diemert, A., Harrison, A., Vignali, M., Sanders, C., Robins, H. S., Tolosa, E., Heesen, C., Arck, P. C., Scheffold, A., Chan, K., Emerson, R. O., Friese, M. A., and Gold, S. M. (2019). T cell repertoire dynamics during pregnancy in multiple sclerosis. *Cell Rep.*, 29(4):810–815.e4.

Ranganath, R., Gerrish, S., and Blei, D. M. (2014). Black box variational inference. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pages 814–822. PMLR.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J., Abbeel, P., and Song, Y. S. (2019). Evaluating protein transfer learning with TAPE. In *Advances in Neural Information Processing Systems*, volume 32, pages 9689–9701.

Ravikumar, A., Arzumanyan, G. A., Obadi, M. K. A., Javanpour, A. A., and Liu, C. C. (2018). Scalable, continuous evolution of genes at mutation rates above genomic error thresholds. *Cell*, 175(7):1946–1957.e13.

Richardson, E. and Weiss, Y. (2018). On GANs and GMMs. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, pages 5847–5858.

Russ, W. P., Figliuzzi, M., Stocker, C., Barrat-Charlaix, P., Socolich, M., Kast, P., Hilvert, D., Monasson, R., Cocco, S., Weigt, M., and Ranganathan, R. (2020). An evolution-based model for designing chorismate mutase enzymes. *Science*, 369:440–445.

Sarkisyan, K. S., Bolotin, D. A., Meer, M. V., Usmanova, D. R., Mishin, A. S., Sharonov, G. V., Ivankov, D. N., Bozhanova, N. G., Baranov, M. S., Soylemez, O., Bogatyreva, N. S., Vlasov, P. K., Egorov, E. S., Logacheva, M. D., Kondrashov, A. S., Chudakov, D. M., Putintseva, E. V., Mamedov, I. Z., Tawfik, D. S., Lukyanov, K. A., and Kondrashov, F. A. (2016). Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401.

Shimko, T. C., Fordyce, P. M., and Orenstein, Y. (2020). DeCoDe: degenerate codon design for complete protein-coding DNA libraries. *Bioinformatics*, 36(11):3357–3364.

Shin, J.-E., Riesselman, A. J., Kollasch, A. W., McMahon, C., Simon, E., Sander, C., Manglik, A., Kruse, A. C., and Marks, D. S. (2021). Protein design and variant prediction using autoregressive generative models. *Nat. Commun.*, 12(1):2403.

Sinai, S., Wang, R., Whatley, A., Slocum, S., Locane, E., and Kelsic, E. D. (2020). AdaLead: A simple and robust adaptive greedy search algorithm for sequence design.

Sriperumbudur, B. K., Fukumizu, K., Gretton, A., Schölkopf, B., and Lanckriet, G. R. G. (2009). On integral probability metrics, φ-divergences and binary classification.

Suzek, B. E., Wang, Y., Huang, H., McGarvey, P. B., Wu, C. H., and UniProt Consortium (2015). UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932.

Tomezsko, P. J., Corbin, V. D. A., Gupta, P., Swaminathan, H., Glasgow, M., Persad, S., Edwards, M. D., Mcintosh, L., Papenfuss, A. T., Emery, A., Swanstrom, R., Zang, T., Lan, T. C. T., Bieniasz, P., Kuritzkes, D. R., Tsibris, A., and Rouskin, S. (2020). Determination of RNA structural diversity and its role in HIV-1 RNA splicing. *Nature*, 582:438–442.

Twist Bioscience (2020). *Combinatorial Variant Libraries.*

Voigt, C. A., Martinez, C., Wang, Z.-G., Mayo, S. L., and Arnold, F. H. (2002). Protein building blocks preserved by recombination. *Nat. Struct. Biol.*, 9(7):553–558.

Weinstein, E. N. and Marks, D. S. (2021). A structured observation distribution for generative biological sequence prediction and forecasting. In *Proceedings of the 38th International Conference on Machine Learning*, 139, pages 11068–11079. PMLR.

Wilson, D. S. and Keefe, A. D. (2001). Random mutagenesis by PCR. *Curr. Protoc. Mol. Biol.*, Chapter 8:Unit8.3.

Yang, K. K., Chen, Y., Lee, A., and Yue, Y. (2019). Batched stochastic bayesian optimization via combinatorial constraints design. In Masashi, Chaudhuri, K. A., editor, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 3410–3419. PMLR.

Zhang, T. (2003). Sequential greedy approximation for certain convex optimization problems. *IEEE Trans. Inf. Theory*, 49(3):682–691.

Supplementary Material: Optimal Design of Stochastic DNA Synthesis Protocols based on Generative Sequence Models

Table S1: Synthesis model notation.

General notation	**Description**
$\mathbb{Z}_+$	The set of positive non-zero integers.
$\mathbb{R}_+$	The set of positive non-zero reals.
Δ_M	The $M-1$ probability simplex.
$(\Delta_M)^D$	The set of matrices with D rows and each row in Δ_M.
e_j	The length 4 vector of all zeros except a 1 at position j.
Hyperparameter	**Description**
$M \in \mathbb{Z}_+$	Number of templates.
$K \in \mathbb{Z}_+$	Number of pools.
$L_k \in \mathbb{Z}_+$	Length (in codons) of templates in pool $k \in \{1, \ldots, K\}$.
$L = \sum_k L_k$	Total length (in codons) of generated sequences.
$A \in \mathbb{Z}_+$	Number of codon or nucleotide mixtures.
$S \in \mathbb{R}_+^4$	Substitution matrix. $S_{b',b}$ is the probability of mutating b to b'.
	$S^\top \in (\Delta_4)^4$ where $S^\top$ is the transpose of S.
	The columns of S are linearly independent.
$T \in \{0,1\}^{64\times 21}$	Translation matrix, mapping from codons
	to the twenty amino acids plus the stop codon.
	$T_{(b_1,b_2,b_3)d} = 1$ if (b_1, b_2, b_3) codes for d,
	and $T_{(b_1,b_2,b_3)d} = 0$ otherwise.
	We assume the standard (universal) codon table.
	Note $\sum_{b_1,b_2,b_3} T_{(b_1,b_2,b_3)d} \geq 1$ for all $d \in \{1, \ldots, 21\}$.
Codon diversification model	**Description**
$\mathcal{U} = \Delta_{64}$	*Arbitrary codon mixtures.*
$\mathcal{U} = \{v_1, \ldots, v_A\}$	*Finite codon mixtures.*
$\mathcal{U} = \{v_1, \ldots, v_A\} \otimes \{v_1, \ldots, v_A\} \otimes \{v_1, \ldots, v_A\}$	*Finite nucleotide mixtures.*
	Nb. in this model, the probability of a codon (b_1, b_2, b_3) is
	the product of mixture probabilities $v_{a_1b_1}v_{a_2b_2}v_{a_3b_3}$
	where $a_1, a_2, a_3 \in \{1, \ldots, A\}$
$\mathcal{U} = \{S^\tau e_1, \ldots, S^\tau e_4\} \otimes \{S^\tau e_1, \ldots, S^\tau e_4\}$	*Enzymatic mutagenesis.*
$\otimes\{S^\tau e_1, \ldots, S^\tau e_4\}$	Nb. in this model, the probability of a codon (b_1, b_2, b_3) is
	the product of mixture probabilities $S^\tau_{b_1a_1}S^\tau_{b_2a_2}S^\tau_{b_3a_3}$
	where $a_1, a_2, a_3 \in \{1, \ldots, 4\}$
Assembly model	**Description**
$Z_{i1} \sim \text{Categorical}(w)$	*Fixed assembly*
$Z_{i2} := \ldots := Z_{iK} := Z_{i1}$	
$Z_{ik} \sim \text{Categorical}(w_k)$ for all $k \in \{1, \ldots, K\}$	*Combinatorial assembly*

Continued on next page...

Continued from previous page...

Parameter	**Description**
w	Template probabilities. $w \in \Delta_M$ if using fixed assembly. $w \in (\Delta_M)^K$ if using combinatorial assembly.
v	Nucleotide or codon mixture probabilities. $v \in (\Delta_{64})^A$ if using finite codon mixtures, $v \in (\Delta_4)^A$ if using finite nucleotide mixtures.
τ	Number of rounds of mutagenesis. $\tau \in \mathbb{Z}_+$.
u	Template defining codon probabilities. $u_{kzj} \in \mathcal{U}$ for all $k \in \{1, \ldots, K\}$, $z \in \{1, \ldots, M\}$ and $j \in \{1, \ldots, L_k\}$
Latent variable	**Description**
$Z_i \in \{1, \ldots, M\}^K$	Templates to generate sequence i. Z_{ik} is the template drawn from pool k.
$C_i \in (\Delta_{64})^L$	Codon probabilities to generate sequence i. $C_{ij(b_1,b_2,b_3)}$ is the probability of generating codon (b_1, b_2, b_3) at position j.
$H_i \in \{0,1\}^{L\times 64}$	Codons of generated sequence i. $H_{ij(b_1,b_2,b_3)} = 1$ if the codon (b_1, b_2, b_3) is at position j, and $H_{ij(b_1,b_2,b_3)} = 0$ otherwise.
Observed variable	**Description**
$X_i \in \{0,1\}^{L\times 21}$	The ith generated protein sequence, one-hot encoded and including the stop codon. $X_{ijd} = 1$ if the amino acid d is at the jth position, and $X_{ijd} = 0$ otherwise.

S1 SOCIETAL IMPACT STATEMENT

Variational synthesis accelerates the application of generative sequence models in the laboratory, and is thus a potentially powerful enabling technology for synthetic biology, and may have a wide range of positive societal impacts. It can be used to improve the scale and accuracy of protein engineering efforts, opening the door to new therapeutics, enzymes and materials. It also may be used in concert with pathogen forecasting models to address problems such as pathogen preparedness, by providing a technique for building large libraries of predicted future sequences. However, synthetic biology is also a dual use technology, with a range of potentially dangerous applications as well; for reviews of dual use concerns, see e.g. El Karoui et al. (2019) and National Research Council et al. (2007).

S2 MODEL DETAILS AND LIMITATIONS

In this section we explain further the synthesis models proposed in Section 2.1, as well some of the limitations of our mathematical idealization.

Physically, for the finite codon or nucleotide mixture models, codon diversification happens during chemical synthesis of oligos (DNA segments). DNA in each well (or isolated reaction volume) is synthesized position by position, with mixtures of nucleotides or codons (trinucleotides) added in defined ratios one at a time, such that a large number of different molecules is eventually constructed. Twist Bioscience's combinatorial variant libraries, which can achieve arbitrary codon mixtures, rely on proprietary technology; however, it produces analogous results (Twist Bioscience, 2020). For all of these technologies, what we refer to as a "template" corresponds physically to a very large number of molecules in an individual well, with independent nucleotide or codon probabilities at each site. We assume that the number of molecules is effectively infinite in comparison to N_1, such that we do not need to account for sampling noise at this stage. We ignore the possibility of skipped positions, where nucleotides or codons randomly fail to add to the growing oligos, a type of error that is sometimes of particular concern for trimer-based synthesis. We enforce the constraint that the number of mixtures A is finite and small, since to the best of our knowledge commercially available technologies have this requirement, but it is not necessarily a fundamental technological constraint (Pazdernik and Bowersox, 2016).

Physically, for the enzymatic mutagenesis model, template oligos are synthesized deterministically, such that there is a large population of identical molecules in each well. Codon diversification occurs only after assembly (i.e. after oligos from different wells are combined) and may take place either *in vitro* or *in vivo*. We assume that there is an error correction mechanism after each round of mutagenesis, such that each strand of each DNA molecule has effectively gone through the same number of rounds of mutagenesis; in some ePCR protocols error correction is not used, and so alternative models may be more appropriate (Moore and Maranas, 2000; Pritchard et al., 2005). We also assume that the mutation probability depends only on individual nucleotides, and not their sequence context, although empirically dependencies on sequence context (especially the adjacent two nucleotides) can be found (Alexandrov and Stratton, 2014). Finally, we require that each template undergoes the same number of rounds of mutagenesis τ, with the same enzyme and thus the same S. For small M, it can be experimentally tractable in many cases to use different τ, and even different S, for each template, in which case the model should be adjusted to make τ and S depend on the template.

Physically, assembly requires joining oligos together using e.g. Gibson assembly (Gibson et al., 2009). For the fixed assembly model, the oligos corresponding to the kth template in each pool must be joined in an isolated reaction, for all $k \in \{1, \ldots, K\}$; in combinatorial assembly, the sets of oligos corresponding to each template in each pool are first mixed, and then oligos from these combined pools are joined. Assembly requires short overhangs, sequences that closely match one another, at the ends of each oligo that are to be joined. Our synthesis model ignores any restrictions that come from overhangs needing to match, as well as variation in assembly probability that depend on overhang mismatch. Our model also assumes full control over the relative concentration of templates, w. While this is tractable for low M, it may be more challenging for large M, particularly if technologies like Dropsynth are used for fixed assembly (Plesa et al., 2018).

S3 OPTIMIZATION DETAILS

S3.1 Exact Solutions

As an example of a target sequence model which we can exactly match, consider a RegressMuE (Weinstein and Marks, 2021), which has been used for forecasting the evolution of influenza. Let $\mathbf{B}$ be a covariate vector (e.g. a future time), let $\mathbf{\Theta}$ be the regression coefficients, and let $\mathbf{W}$ be the latent alignment. The predictive distribution $p(x|\mathbf{B}, \mathbf{W}, \mathbf{\Theta})$ can be written as Categorical($\mathbf{U}$), where $\mathbf{U}$ is a matrix of independent amino acid probabilities over L positions. We can exactly match this distribution with a synthesis model using $M = 1$ templates, fixed assembly and arbitrary codon mixtures.

We can also approximate the posterior predictive distribution. Let $p(\mathbf{\Theta}|\mathcal{D})$ be the posterior distribution over regression parameters given the training data. The posterior predictive distribution can be approximated as $\sum_{m=1}^{M} \frac{1}{M} p(x|\mathbf{B}, \mathbf{W}, \mathbf{\Theta}_m)$ where $\mathbf{\Theta}_1, \ldots, \mathbf{\Theta}_M \sim p(\mathbf{\Theta}|\mathcal{D})$ are posterior samples. This distribution can be exactly matched by a stochastic synthesis model using fixed assembly with $w = (\frac{1}{M}, \ldots, \frac{1}{M})$ and arbitrary codon mixtures.

S3.2 Stochastic EM

We used the online EM algorithm proposed by Cappé and Moulines (2008), modified to update using minibatches instead of individual datapoints. Here we derive the algorithm for the stochastic synthesis model (Equation 1). Without loss of generality, we focus on combinatorial assembly models; the fixed assembly case can be obtained by setting $K = 1$. The local variable of the synthesis model is Z_i, which we represent here as a one-hot encoding, i.e. $Z_i \in \{0, 1\}^{K \times M}$. At iteration t of the optimization algorithm, given the current parameter estimate $\theta^{(t)} = (w^{(t)}, u^{(t)}, v^{(t)}, \tau^{(t)})$, the conditional expectation of Z_i can be written as

$$r_{ikm} := \mathbb{E}_{q_{\theta^{(t)}}}[Z_{ikm}|X_i] = \frac{w_{k,m} \exp\left(\sum_{j=1}^{L_k} \log(u_{kmj} \cdot T) \cdot X_{i(j+\bar{L}_k)}\right)}{\sum_{m'=1}^{M} w_{k,m'} \exp\left(\sum_{j=1}^{L_k} \log(u_{km'j} \cdot T) \cdot X_{i(j+\bar{L}_k)}\right)}, \tag{S1}$$

where $\bar{L}_k = \sum_{k'<k} L_k$. Now we can compute the conditional expectation of the mean log likelihood as

$$\begin{aligned} Q_{\theta^{(t)}}(X_1, \ldots, X_N; \theta) &:= \frac{1}{N}\mathbb{E}_{q_{\theta^{(t)}}}[\log q_\theta(X_1, \ldots, X_N, Z_1, \ldots, Z_N)] \\ &= \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M}\left[\sum_{j=1}^{L_k}\log(u_{kmj}\cdot T)\cdot X_{i(j+\bar{L}_k)}r_{ikm} + \log w_{km}r_{ikm}\right]. \end{aligned} \tag{S2}$$

In standard EM, we would optimize this function with respect to θ. However, this requires summing over the whole dataset at each step. To derive the stochastic EM algorithm, we rewrite $Q_{\theta^{(t)}}$ in terms of summary statistics of the data that can be estimated from minibatches. In particular, let $\mathcal{S} \subseteq \{1, \ldots, N\}$ be a subset of the data, and define the summary statistics

$$\begin{aligned} \bar{s}^{(1)}(X_\mathcal{S}; \theta^{(t)})_{kmj} &:= \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} X_{ij}r_{ikm}, \\ \bar{s}^{(2)}(X_\mathcal{S}; \theta^{(t)})_{km} &:= \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} r_{ikm}. \end{aligned} \tag{S3}$$

Now we can estimate $Q_{\theta^{(t)}}$ as

$$\hat{Q}(\bar{s}; \theta) := \sum_{k=1}^{K}\sum_{m=1}^{M}\left[\sum_{j=1}^{L_k}\log(u_{kmj}\cdot T)\cdot \bar{s}^{(1)}_{km(j+\bar{L}_k)} + \log w_{km}\bar{s}^{(2)}_{km}\right]. \tag{S4}$$

The complete algorithm alternates between estimating summary statistics from minibatches of data $\mathcal{S}^{(t)}$ drawn at each step and maximizing the estimated expected log likelihood $\hat{Q}_{\theta^{(t)}}$,

$$\begin{aligned} \hat{s}^{(t+1)} &= \hat{s}^{(t)} + \gamma^{(t+1)}(\bar{s}(X_{\mathcal{S}^{(t)}}; \theta^{(t)}) - \hat{s}^{(t)}) \\ \theta^{(t+1)} &= \operatorname*{argmax}_\theta \hat{Q}(\hat{s}^{(t+1)}; \theta) \end{aligned} \tag{S5}$$

where $\gamma^{(t)}$ is the step size. As suggested by Cappé and Moulines (2008), we set $\gamma^{(t)} = t^{-0.6}$. We also use Polyak-Ruppert averaging, as suggested by Cappé and Moulines (2008), taking the mean of the summary statistics $\hat{s}^{(t)}$ for the last half of training, i.e. $\hat{s}^* = \frac{2}{t_{\max}}\sum_{t=(t_{\max}/2+1)}^{t_{\max}}\hat{s}^{(t)}$, and producing the final parameter estimate $\hat{\theta}^* = \mathrm{argmax}_\theta \hat{Q}(\hat{s}^*; \theta)$.

The maximization step $\theta^{(t+1)} = \mathrm{argmax}\, \hat{Q}(\hat{s}^{(t)}; \theta)$ can vary depending on the codon diversification technology used. For all technologies, we have

$$w^{(t+1)} = \hat{s}^{(t+1)(2)}_{km}. \tag{S6}$$

For arbitrary codon mixtures and finite codon mixtures, we can without loss of generality pick one codon for each amino acid and the stop symbol, and work with template probabilities $\tilde{u}$ directly over amino acids, i.e. where $\tilde{u}_{k,m,j,d}$ is the probability of amino acid d at position j of template m in pool k. Then, for arbitrary codon mixtures,

$$\tilde{u}^{(t+1)}_{kmjd} = \frac{\bar{s}^{(t+1)(1)}_{km(j+\bar{L}_k)d}}{\sum_{d'=1}^{21}\bar{s}^{(t+1)(1)}_{km(j+\bar{L}_k)d'}}. \tag{S7}$$

For finite codon mixtures, let $\tilde{\chi}_{kmj}$ be a one-hot encoding of the codon mixture used at position j of template m in pool k, such that $\tilde{\chi}_{kmj} \in \{0,1\}^A$. We work directly with mixtures defined over amino acids, with $\tilde{v}_{ad}$ the probability of amino acid d in mixture a. Thus $\tilde{u}_{kmj} = \tilde{\chi}_{kmj}\cdot\tilde{v}$. Then we can use the coordinate-wise update

$$\begin{aligned} \tilde{\chi}^{(t+1)}_{kmj} &= \mathrm{argmax}_a \sum_{d=1}^{21}\log(\tilde{v}_{ad})\hat{s}^{(t+1)(1)}_{km(j+\bar{L}_k)d} \\ \tilde{v}^{(t+1)}_{ad} &= \frac{\sum_{k=1}^{K}\sum_{m=1}^{M}\sum_{j=1}^{L_k}\hat{s}^{(t+1)(1)}_{km(j+\bar{L}_k)d}\tilde{\chi}^{(t+1)}_{kmja}}{\sum_{d'=1}^{21}\sum_{k=1}^{K}\sum_{m=1}^{M}\sum_{j=1}^{L_k}\hat{s}^{(t+1)(1)}_{km(j+\bar{L}_k)d'}\tilde{\chi}^{(t+1)}_{kmja}} \end{aligned} \tag{S8}$$

For finite nucleotide mixtures, we use χ_{kmj1} to denote a one-hot encoding of the mixture used at the first position of the codon at position j in template m in pool k, i.e. $\chi_{kmj1} \in \{0,1\}^A$, and likewise for χ_{kmj2} and χ_{kmj3}. We update χ by optimizing over all three positions of each codon jointly, enumerating all combinations of a_1, a_2 and a_3,

$$\chi^{(t+1)}_{kmj} = \underset{(a_1,a_2,a_3)}{\text{argmax}} \sum_{d=1}^{21} \log\Big(\sum_{b_1,b_2,b_3} v_{a_1 b_1} v_{a_2 b_2} v_{a_3 b_3} T_{(b_1,b_2,b_3)d}\Big) \bar{s}^{(t+1)(1)}_{km(j+\bar{L}_k)d}. \tag{S9}$$

Once χ has been updated, we update v. This is harder, as there is no closed form solution. We directly optimize $\hat{Q}$ with respect to v by taking gradients and applying 5 steps of the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.01 (that is, we take 5 steps of Adam for every 1 EM update). For enzymatic mutagenesis, we can also apply Equation S9 to update χ, replacing v with S^τ. To update τ, we directly enumerate all values of $\hat{Q}$ for $\tau \in \{1, \ldots, \tau_{\max}\}$ and choose the maximum.

Code implementing the stochastic EM algorithm for all of the proposed stochastic synthesis models is available in the Supplementary Material.

S3.3 Choosing $\tilde{N}$

Recall that our proposed black-box optimization procedure is to draw $X_1, \ldots, X_{\tilde{N}} \sim p$ computationally and then maximize the synthesis model parameters,

$$\hat{\theta}_{\tilde{N}} := \text{argmax}_\theta \sum_{i=1}^{\tilde{N}} \log q_\theta(X_i). \tag{S10}$$

In this section, we argue that $\tilde{N}$ should be chosen to be either equal to N_1, or, if N_1 is too large to be tractable computationally, $\tilde{N}$ should be as large as is tractable. In particular, we *do not* suggest choosing $\tilde{N}$ to be larger than N_1, nor do we suggest regularizing θ as one would in a standard inference problem. The reason is that "overfitting" the synthesis model to the samples $X_1, \ldots, X_{\tilde{N}}$ can help rather than hurt.

To be more precise, consider the extreme case where q_θ can exactly match the empirical distribution of $X_1, \ldots, X_{N_1} \sim p$ but cannot exactly match p itself. For example, this situation can occur when using fixed assembly and $M = N_1$, allowing each mixture component be a point mass. If we use $\tilde{N} = N_1$, we find

$$q_{\hat{\theta}_{\tilde{N}}}(x) = \frac{1}{\tilde{N}} \sum_{i=1}^{\tilde{N}} \delta_{X_i}(x) \tag{S11}$$

where $\delta_{x'}(x)$ is the Kronecker delta function at x'. In this case, variational synthesis is equivalent to large-scale MC synthesis, and will produce N_1 samples from p.[3] On the other hand, if we let $\tilde{N} \to \infty$, we have $\hat{\theta}_{\tilde{N}} \to \theta^*$. In this case, variational synthesis will produce N_1 samples from $q_{\theta^*} \neq p$. Thus, it can be preferable to use $\tilde{N} = N_1$ as compared to $\tilde{N} > N_1$, since using $\tilde{N} = N_1$ leads to synthesis of N_1 exact samples from p instead of N_1 samples from $q_{\theta^*} \neq p$.

In practice, of course, q_θ will rarely be able to exactly match the empirical distribution of samples from p. Nonetheless, we expect using $\tilde{N} \approx N_1$ to be useful, as in this case we avoid trying to match $q_{\hat{\theta}_{\tilde{N}}}$ to components of p that are too rare to occur in practice, and instead regularize $q_{\hat{\theta}_{\tilde{N}}}$ towards the empirical distribution of samples from p.

S3.4 Variable Length Protein Sequences

To handle variable length protein sequences, we treat everything past the stop codon as missing data which does not contribute to the likelihood. That is, for a sequence X_i with a stop codon at position $\mathbf{j}$, we have $q_\theta(X_i) = q_\theta(X_{i,1:\mathbf{j}})$.

[3]Technically, variational synthesis in this case produces a size N_1 bootstrap of N_1 samples from p, rather than directly producing N_1 samples from p. Although bootstrapping introduces some additional sampling noise, we expect it is unlikely in practice to make using $q_{\hat{\theta}_{\tilde{N}}}$ worse than using q_{θ^*}, since the bootstrap directly approximates p. Section S5.1 discusses this subtlety further.

S4 RELATED WORK DETAILS

S4.1 DeCoDe

DeCoDe can be applied to datasets of fixed-length (or aligned) sequences, $X'_1, \ldots, X_{N'}$, which are assumed to be unique (i.e. $X'_i \neq X'_{i'}$ if $i \neq i'$). Consider the empirical distribution $p(x) = \sum_{i=1}^{N'} \delta_{X'_i}(x)$ where $\delta_{x'}(x)$ is the Kronecker delta. Take q_θ to be a stochastic synthesis model using finite nucleotide mixtures and fixed assembly, with $\theta = (w, u, v)$. Let $\text{supp}(p)$ denote the support of p, i.e. the set of all length L sequences with non-zero probability. Let $\zeta \in \mathbb{Z}_+$ denote the maximum allowed support of q_θ. Then, we can rewrite the DeCoDe objective (Section 2.2.2 in Shimko et al. (2020)) in terms of the size of the intersection of supports of p and q_θ,

$$\theta^* = \underset{\theta:\text{supp}(q_\theta)\leq\zeta}{\text{argmax}} \; |\text{supp}(p) \cap \text{supp}(q_\theta)|. \tag{S12}$$

Note that the size of the intersection of supports does not correspond to a valid divergence between p and q_θ.

S4.2 SCHEMA

RASPP (Endelman et al., 2004) is an algorithm for designing site-directed recombination or combinatorial assembly libraries based on a crystal structure and a dataset of homologous proteins from the same family. It chooses a set of template lengths $L_1, \ldots, L_K$, where $L_{\min} \leq L_k \leq L_{\max}$ for $k \in \{1, \ldots, K\}$, in order to minimize the SCHEMA score, roughly the number of structural contacts between positions of the protein generated by different template pools. In this section we give a heuristic argument connecting RASPP to variational synthesis, in the special case where RASPP finds a solution with no structural contacts across regions covered by each pool.

Consider a target model p that consists of a Potts model learned from the same protein family as the dataset of homologous proteins. In general, the Potts model will infer energetic interactions only between positions of the alignment that are in structural contact (Marks et al., 2011). Let $\tilde{L}_k$ denote the region generated by template k, i.e. $\tilde{L}_1 = \{1, \ldots, L_1\}$, $\tilde{L}_2 = \{L_1 + 1, \ldots, L_1 + L_2\}$, etc. and let $p(x_{\tilde{L}_k})$ denote the marginal of p over these positions. For the set of $L_1, \ldots, L_k$ chosen by RASPP, we have no structural contacts across regions, and so no energetic interactions under the Potts model, and thus $p(x_{\tilde{L}_k}, x_{\tilde{L}_{k'}}) = p(x_{\tilde{L}_k})p(x_{\tilde{L}_{k'}})$ for $k \neq k'$. In other words, there is no correlation between segments under the Potts model p. When using stochastic synthesis with combinatorial assembly, there is also no correlation between segments under q_θ. If we try to minimize the KL divergence between a q_θ with combinatorial assembly and the Potts model p, and optimize the template lengths $L_1, \ldots, L_K$, we can expect in general to find a similar solution to RASPP, where both the SCHEMA score and the correlation between templates under p is zero.

S5 THEORY DETAILS

Note that the proofs in this section rely on the definitions in Table S1.

S5.1 The MC Synthesis Estimator

In our theoretical analysis we do not treat MC synthesis as variational synthesis with point mass (deterministic) mixture components. In particular, we analyze the estimator

$$\begin{aligned} &X_1, \ldots, X_{N_0} \sim p, \\ &\hat{I}^{(a)} := \frac{1}{N_0}\sum_{i=1}^{N_0} f(X_i), \end{aligned} \tag{S13}$$

which comes from measuring each synthesized sequence individually, and *not* the alternative estimator

$$\begin{aligned} &X'_1, \ldots, X'_{N_0} \sim p, \\ &X_1, \ldots, X_{N_1} \sim \frac{1}{N_0}\sum_{i=1}^{N_0} \delta_{X'_i}(x), \\ &\hat{I'}^{(a)} := \frac{1}{N_1}\sum_{i=1}^{N_1} f(X_i). \end{aligned} \tag{S14}$$

which would come from pooling the synthesized sequences and then measuring a random sample of size N_1 (here $\delta_{x'}(x)$ is the Kronecker delta function at x'). Note this alternative estimator $\hat{I}'^{(a)}$ takes the form of a bootstrap estimator of size N_1, taken from an initial sample of size N_0 from p, and thus in general introduces additional sampling noise as compared to $\hat{I}^{(a)}$. There are three reasons for focusing our analysis on $\hat{I}^{(a)}$ instead of $\hat{I}'^{(a)}$. First, since N_0 is low, in practice it is often tractable for experimentalists to measure the N_0 sequences individually (e.g. in 96 well plates), rather than pooling them, making the estimate $\hat{I}^{(a)}$ possible. Second, in the limit where N_1 is much greater than N_0, the estimators converge, making $\hat{I}^{(a)}$ a reasonable approximation for pooled experiments in practice. Third, we want our analysis to be conservative in measuring the benefits of variational synthesis vis-à-vis the alternative, MC synthesis, so we use the better estimator $\hat{I}^{(a)}$.

S5.2 Proof of Proposition 4.1

Proof. Using Jensen's inequality,

$$\frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}\mathbb{E}[|\hat{I}^{(a)}-I|] \leq \frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}\sqrt{\frac{1}{N_0^2}\mathbb{E}_p\left[\left(\sum_{i=1}^{N_0}(f(X_i)-\mathbb{E}_p[f(X)])\right)^2\right]} \leq \frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}\sqrt{\frac{\mathbb{V}_p[f(X)]}{N_0}} \leq \frac{1}{\sqrt{N_0}} \tag{S15}$$

where $\mathbb{V}_p[f(x)]$ is the variance with respect to p.

We can decompose the error in the $\hat{I}^{(b)}$ estimate into variance and bias terms, and then apply a similar analysis.

$$\frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}\mathbb{E}[|\hat{I}^{(b)}-I|] \leq \frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}\mathbb{E}[|\hat{I}^{(b)}-\mathbb{E}_{q_{\theta^*}}[f(X)]|]+\frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}|\mathbb{E}_{q_{\theta^*}}[f(X)]-\mathbb{E}_p[f(X)]| \leq \frac{1}{\sqrt{N_1}}+\text{TV}(p,q_{\theta^*}). \tag{S16}$$

where we have used the integral probability metric representation of the total variation metric $\text{TV}(\cdot,\cdot)$ (Sriperumbudur et al., 2009). The result follows from application of Pinsker's inequality. $\square$

We can see from the proof that the bound in Equation 3 could be tighter if we use total variation in place of KL. It could also be tighter if we restrict the family of functions $\mathcal{F}$ further. In particular, consider the metric space defined over the set of fixed length discrete sequences $\mathcal{X}$ with the Hamming distance $\|x-x'\|_H := \sum_{j=1}^{L}\sum_{d=1}^{21}\frac{1}{2}|x_{jd}-x'_{jd}|$ (where x is a one hot encoding of a length L nucleotide sequence). Then, we can introduce the function family $\mathcal{F}_W := \{f : \|f\|_L \leq D_{\max}\}$, that is, the set of functions with bounded Lipschitz constant $\|f\|_L := \max_{x,x'\in\mathcal{X},x\neq x'}|f(x)-f(x')|/\|x-x'\|_H$. Biologically, the Lipschitz constant is interpretable as the sensitivity of a sequence's biological function to single mutations. In particular, if a point mutation can dramatically change the assayed property of the sequence, then the Lipschitz constant will be large; otherwise it will be small. If we assume the Lipschitz constant of the experimental assay is bounded by some constant $D_{\max}$, we can find an alternative error bound on the stochastic synthesis estimator:

$$\frac{1}{f_{\max}}\sup_{f\in\mathcal{F}\cap\mathcal{F}_W}\mathbb{E}[|\hat{I}^{(b)}-I|] \leq \frac{1}{\sqrt{N_1}}+\frac{D_{\max}}{f_{\max}}\text{W}(p,q_{\theta^*}). \tag{S17}$$

where $\text{W}(p,q) := \inf_{\gamma\in\Gamma(p,q)}\int\|x-x'\|_H\gamma(x,x')$ is the first Wasserstein distance, with $\Gamma(p,q)$ the set of couplings of p and q. This result follows from Equation S16 by applying the Kantorovich-Rubinstein duality theorem (e.g. Dudley (2002), Theorem 11.8.2), using the fact that the metric space of finite sequences with the Hadamard distance is a finite discrete space and separable. We see from Equation S17 that the error bound on variational synthesis can be lower than that in Equation S16, so long as $D_{\max}$ is sufficiently small. In other words, we can get away with using synthesis models that do not match p closely if the assay is not very sensitive to small changes in sequence.

S5.3 Importance Sampling Estimates

In some cases we can get access to paired sequence and function data, and in particular the dataset $\mathcal{D}_0 := \{(f(X_i), X_i) : f(X_i) \neq 0\}$. For instance, if we deep sequence the hits of a screen, with $f : \mathcal{X} \mapsto \{0,1\}$, we will have $\mathcal{D}_0 := \{(1, X_i) : f(X_i) = 1\}$. We can then construct an importance-sampling estimate of $I = \mathbb{E}_p[f(X)]$

using samples $X_1, \ldots, X_{N_1} \sim q_{\theta^*}$,

$$\hat{I}^{(c)} := \frac{1}{N_1}\sum_{i=1}^{N_1} f(X_i)\frac{p(X_i)}{q_{\theta^*}(X_i)} = \frac{1}{N_1}\sum_{X_i \in \mathcal{D}_0} f(X_i)\frac{p(X_i)}{q_{\theta^*}(X_i)}.$$

Unlike $\hat{I}^{(b)}$, this estimator is unbiased: $\mathbb{E}_{q_{\theta^*}}[f(X)p(X)/q_{\theta^*}(X)] = I$. However, $\hat{I}^{(c)}$ still takes advantage of a large number of samples, making possible lower variance than $\hat{I}^{(a)}$. In particular, we have

$$\frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}\mathbb{E}_{q_{\theta^*}}[|\hat{I}^{(c)} - I|] \le \frac{1}{\sqrt{N_1}}\sqrt{\textsc{chi}(p||q_{\theta^*})} \tag{S18}$$

where CHI is the chi divergence, which can be defined as $\textsc{chi}(p||q) = \mathbb{V}_q[\frac{p(X)}{q(X)}]$. We can derive this result following the same analysis as in Equation S15,

$$\frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}\mathbb{E}_{q_{\theta^*}}[|\hat{I}^{(c)} - I|] \le \frac{1}{f_{\max}}\sup_{f\in\mathcal{F}}\sqrt{\frac{\mathbb{V}_{q_{\theta^*}}[f(X)\frac{p(X)}{q_{\theta^*}(X)}]}{N_1}} \le \frac{1}{\sqrt{N_1}}\sqrt{\textsc{chi}(p||q_{\theta^*})}. \tag{S19}$$

Note that our suggested black-box optimization procedure for variational synthesis (Section 2.2) is intended to help ensure high discovery rates (maximizing $\sum_{i=1}^{N_1} f(X_i)$) but not to ensure accurate importance sampling estimates. In particular, the KL divergence does not provide a particularly tight bound on the CHI divergence (see e.g. Proposition 2 in Dragomir (1999)), so it is likely preferable to (if possible) directly optimize the CHI divergence (Dieng et al., 2017).

S5.4 Proof of Corollary 4.2

Proof. We have

$$\mathbb{E}[N_1\hat{I}^{(b)} - N_0\hat{I}^{(a)}] \ge (N_1 - N_0)I - N_1\sup_{f\in\mathcal{F}}\mathbb{E}[|\hat{I}^{(b)} - I|] - N_0\sup_{f\in\mathcal{F}}\mathbb{E}[|\hat{I}^{(a)} - I|] \tag{S20}$$

Applying Proposition 4.1 yields the result. □

S5.5 Proof of Proposition 4.3

Before proving Proposition 4.3, we first prove a lemma that shows – as long as we are not using enzymatic mutagenesis – that we can construct templates that are arbitrarily close to a point mass while still having full support. We use $q_\theta(x|c)$ as shorthand for $q_\theta(x|C_i = c)$, and $\delta_{x'}(x)$ to denote the Kronecker delta function which takes value 1 if $x = x'$ and 0 otherwise.

Lemma S5.1. *Assume we are using* ***arbitrary codon mixtures, finite codon mixtures*** *(with $A \ge 21$), or* ***finite nucleotide mixtures*** *(with $A \ge 4$). For any $\epsilon > 0$ sufficiently small, there exists some v such that: for all $\bar{x} \in \mathcal{X}$ there exists a $\bar{c}(\bar{x}) \in \mathcal{U}^L$ such that*

$$q(\bar{x}|\bar{c}(\bar{x})) \ge 1 - \rho L\epsilon \tag{S21}$$

where ρ is a positive constant, and $\operatorname{supp}(q(x|\bar{c}(\bar{x}))) = \mathcal{X}$. In particular, for arbitrary or finite codon mixtures, $\rho = 1$, while for finite nucleotide mixtures, $\rho = 3$.

Proof. We start with the finite codon mixtures case; note that this immediately implies the arbitrary codon mixture case, since the space $\mathcal{U}$ for finite codon mixtures is a subset of the space $\mathcal{U}$ for arbitrary codon mixtures. We choose (arbitrarily) one codon for each amino acid and the stop symbol, and work with mixtures v over these 21 codons (setting the probability of all others to zero). For all $d \in \{1, \ldots, 21\}$, let $v_d = \mathbf{1}_{21}\frac{\epsilon}{21} + e_d^{(21)}(1-\epsilon)$ where $\mathbf{1}_D$ is the length D vector of all ones and $e_d^{(D)}$ is the length D vector of all zeros except a one at position d. Let $\ell(\bar{x})$ be the length of a protein sequence $\bar{x}$.[4] Given $\bar{x}$ we define the $L \times 21$ matrix $\bar{c}(\bar{x}) = \text{concatenate}(v_{\bar{x}_1}, \ldots, v_{\bar{x}_{\ell(\bar{x})}}, v_1, \ldots, v_1)$. Now note that

$$q(\bar{x}|\bar{c}(\bar{x})) \ge (1-\epsilon)^{\ell(\bar{x})} \ge 1 - L\epsilon. \tag{S22}$$

[4]Length is measured up to (and including) the first stop codon or L, whichever comes first.

Next we consider the finite nucleotide mixtures case, which works similarly. For all $b \in \{1, \ldots, 4\}$, let $v_b = \mathbf{1}_4 \frac{\epsilon}{4} + e_b^{(4)}(1-\epsilon)$. Given a protein sequence $\bar{x}$, choose a particular codon for each amino acid and the stop symbol. This defines a DNA sequence $\tilde{x}$, where $\tilde{x}_{j1}$ is the nucleotide in the first position of the codon for the amino acid at position j of $\bar{x}$, and likewise for $\tilde{x}_{j2}$ and $\tilde{x}_{j3}$. We can then choose nucleotide mixtures for each position of a template to match $\tilde{x}$, that is,

$$\bar{c}(\bar{x}) = \text{concatenate}(v_{\tilde{x}_{11}} \otimes v_{\tilde{x}_{12}} \otimes v_{\tilde{x}_{13}}, \ldots, v_{\tilde{x}_{\ell(\bar{x})1}} \otimes v_{\tilde{x}_{\ell(\bar{x})2}} \otimes v_{\tilde{x}_{\ell(\bar{x})3}}, v_1 \otimes v_1 \otimes v_1, \ldots, v_1 \otimes v_1 \otimes v_1).$$

Now we have

$$q(\bar{x}|\bar{c}(\bar{x})) \geq (1-\epsilon)^{3\ell(\bar{x})} \geq 1 - 3L\epsilon. \tag{S23}$$

□

We are now ready to prove Part 1 of Proposition 4.3. The basic idea is to construct a synthesis distribution q_{θ^*} that closely approximates p by convolving with p templates that are approximate point masses.

Part 1 of Proposition 4.3: *When using either **arbitrary codon mixtures**, **finite codon mixtures** (with $A \geq 21$), or **finite nucleotide mixtures** (with $A \geq 4$): for any $p \in \mathcal{P}(\mathcal{X})$ and $\eta > 0$ there exists some M and θ such that (1) $\text{KL}(p\|q_\theta) < \eta$ and (2) $\text{supp}(q_\theta(x|z)) = \mathcal{X}$ for all $z \in \{1, \ldots, M\}$.*

Proof. Let $M = |\mathcal{X}|$, that is, set the number of templates equal to the total number of sequences of length less than or equal to L. Since $\mathcal{X}$ is finite, we can construct for any $\epsilon > 0$ the synthesis distribution $q_\theta(x) = \mathbb{E}_{\bar{X}\sim p}[q(x|\bar{c}(\bar{X}))]$. In this synthesis distribution, the weights w of each mixture component are set by $p(x)$, and $\text{supp}(q_\theta(x|z)) = \mathcal{X}$ for all z by the construction of $\bar{c}$. We now have, applying Lemma S5.1,

$$\begin{aligned}
\text{KL}(p\|q_\theta) &= \sum_{x\in\mathcal{X}} p(x)\log p(x) - \sum_{x\in\mathcal{X}} p(x) \log \big[\sum_{\bar{x}\in\mathcal{X}} q(x|\bar{c}(\bar{x}))p(\bar{x})\big] \\
&\leq \sum_{x\in\mathcal{X}} p(x)\log p(x) - \sum_{x\in\mathcal{X}} p(x) \log \big[q(x|\bar{c}(x))p(x)\big] \\
&\leq \sum_{x\in\mathcal{X}} p(x)\log p(x) - \sum_{x\in\mathcal{X}} p(x) \log \big[(1-\rho L\epsilon)p(x)\big] \\
&\leq -\log(1-\rho L\epsilon)
\end{aligned} \tag{S24}$$

Thus we can choose ϵ sufficiently small that $\text{KL}(p\|q_\theta) < \eta$. □

One concerning aspect of this proof, practically, is that it requires very large M to form the approximation q_θ. How well can we do with smaller M? Combining Theorem 4.2 of Zhang (2003) with the above result, we can say that for any $\eta > 0$ there exists an $\epsilon > 0$ such that $\text{KL}(p\|q_{\theta^*})$ converges to a value less than η at a $1/M$ rate. Note also that our setup differs from the more common case where a mixture model is used for density estimation based on finite data, since we can sample as much as we want from p. We therefore do not analyze the mismatch between a target p and model q_θ that may be caused by finite data.

Next, we prove the second part of Proposition 4.3, showing that enzymatic mutagenesis can fail to approximate arbitrary targets p. The basic idea is that when using enzymatic mutagenesis, the probability of a particular sequence cannot get arbitrarily close to 1, and so the KL divergence between p and q_θ cannot get arbitrarily close to 0.

Part 2 of Proposition 4.3: *When using **enzymatic mutagenesis**: there exists some $p \in \mathcal{P}(\mathcal{X})$ and $\eta > 0$ such that for all M and θ, we have $\text{KL}(p\|q_\theta) > \eta$.*

Proof. Since $\tau > 0$, and the entries of S are all positive, we can see that we are limited in how much mass an enzymatic mutagenesis model can concentrate on just one sequence, i.e.

$$\sup_{\tau>0, c\in\mathcal{U}^L} \sup_{x\in\mathcal{X}} q(x|c,\tau) < 1. \tag{S25}$$

Choose $p(x) = \delta_{x'}(x)$ for some sequence $x' \in \mathcal{X}$, and let q_θ be an enzymatic mutagenesis synthesis model with M templates. Then,

$$\inf_\theta \text{KL}(p\|q_\theta) \geq -\log \sup_{\tau>0, c\in\mathcal{U}^L} \sup_{x\in\mathcal{X}} q(x|c,\tau) > 0. \tag{S26}$$

□

S5.6 Proof of Proposition 4.4

Proof. Let $\tilde{L}_k$ denote the subset of positions generated by template k, i.e. $\tilde{L}_1 = \{1, \ldots, L_1\}$, $\tilde{L}_2 = \{L_1 + 1, \ldots, L_1 + L_2\}$, ... Let $p(x_{\tilde{L}_k})$ denote the marginal of p over these positions. We have, since templates are drawn independently from each pool, $q_\theta(x) = \prod_{k=1}^{K} q_\theta(x_{\tilde{L}_k})$, and so

$$\begin{aligned}
\mathrm{KL}(p\|q_\theta) &= \sum_x p(x)\log p(x) - \sum_{k=1}^{K}\sum_{x_{\tilde{L}_k}} p(x_{\tilde{L}_k})\log q_\theta(x_{\tilde{L}_k}) \\
&= \sum_x p(x)\log p(x) - \sum_{k=1}^{K}\sum_{x_{\tilde{L}_k}} p(x_{\tilde{L}_k})\log p(x_{\tilde{L}_k}) + \sum_{k=1}^{K}\mathrm{KL}(p(x_{\tilde{L}_k})\|q_\theta(x_{\tilde{L}_k})) \qquad \text{(S27)} \\
&\geq \mathrm{KL}(p\|\prod_{k=1}^{K} p(x_{\tilde{L}_k})).
\end{aligned}$$

There exists p for which $\mathrm{KL}(p\|\prod_{k=1}^{K} p(x_{\tilde{L}_k})) > 0$, in particular any p for which there is correlation between templates, proving the result. □

S6 RESULTS DETAILS

S6.1 Datasets and Target Models

S6.1.1 DHFR

We used a dataset of 3,629 sequences in the DHFR family collected using jackhmmer (Eddy, 2011) from the Uniref100 dataset (Suzek et al., 2015), and available as an example dataset from `https://github.com/debbiemarkslab/plmc/tree/master/example/protein/DHFR.a2m`. The multiple sequence alignment has a width of $L = 171$ amino acids. We trained a Potts model using pseudolikelihood maximization as in Hopf et al. (2017), using the plmc package with the default hyperparameters `https://github.com/debbiemarkslab/plmc`. Gaps in the alignment were treated as missing data (not as separate symbols), following the default settings of plmc. The trained Potts model was the target p. We sampled sequences from p using Gibbs sampling, drawing 100,000 samples using 10 parallel chains with a burn-in of 200 steps per chain.

For the analysis of unaligned sequences (Figures 3D and S5), we used the training dataset of 3,629 evolutionary sequences, with gap symbols excluded and stop symbols appended. We refer to this dataset as "DHFR raw".

S6.1.2 GFP

We constructed a dataset of 722 sequences in the GFP family using jackhmmer and UniprotKB (07/2021) (Potter et al., 2018), starting from the seed sequence `GFP_AEQVI` with F64L (a stabilized variant used by Sarkisyan et al. (2016)), with a threshold of 0.3 bit score per residue. We trained an ICA model with a MuE output (Weinstein and Marks, 2021), which is available as an example in the Pyro probabilistic programming language (Bingham et al., 2019) at `https://pyro.ai/examples/mue_factor.html`. The ICA model is similar to a probabilistic PCA model, but uses a Laplace prior on the latent variable instead of a Gaussian; the MuE output uses the default profile HMM-based architecture described in Weinstein and Marks (2021). We used 2 latent dimensions in the ICA model, a latent sequence length of 237 in the MuE, and default priors. The model was trained with stochastic variational inference, with a learning rate of 0.005 and batch size of 5 over 70 epochs, annealing the prior KL divergence linearly over 35 epochs. Using 20% of the data as heldout validation, the model achieves a per residue perplexity on the training set of 3.1 and on the test set of 4.6.

We used the ICA-MuE model to construct a target distribution p. In particular, let ψ be the latent alignment variable of the MuE (the state variable of the Markov chain). We estimated the maximum *a posteriori* value of ψ for the stabilized wild-type GFP (`GFP_AEQVI` with F64L), and then sampled new sequences conditional on this value $\hat{\psi}_{\mathrm{ref}}$ – note that this procedure is a very weak form of supervision, since the stabilized wild-type is known to be functional and produce fluorescence. To limit the diversity of the library relative to the training data, we sampled from the posterior predictive over the latent representation given the observed data, rather than the prior. Explicitly, let $p_{\text{MuE}}(x|\psi,\kappa)$ denote the distribution of the learned ICA-MuE model conditional on the latent alignment ψ and latent representation κ. Let $X'_1,\ldots,X'_{N'}$ denote the training data, and let $p(\kappa|X')$ denote the posterior over the latent representation of a datapoint X' (which can be approximated by the encoder/guide network). The complete generative process p is then defined as

$$\begin{aligned}\kappa_i &\sim \frac{1}{N'}\sum_{i=1}^{N'} p(\kappa|X'_i)\\ X_i &\sim p_{\text{MuE}}(x|\hat{\psi}_{\text{ref}},\kappa_i).\end{aligned} \tag{S28}$$

An important feature of this model is that we are *not* sampling from the conditional distribution of κ given $\hat{\psi}_{\text{ref}}$, that is, we are not sampling sequences with similar latent alignments. Unlike autoregressive models, for example, MuE models allow variation in sequence length and latent alignment to be treated as independent of variation at conserved sites. Thus, although the sequences generated from p are all of the same length, the pattern of amino acids at conserved sites reflects the full diversity of the dataset. Finally, note that in the jackhmmmer constructed-dataset, the first residue (M) of the wild-type sequence `GFP_AEQVI` was not included in the profile HMM envelope, but the sequence-to-function predictor expects this position to be included; we therefore prepended an M to each generated sequence, for a total length of $L = 238$.

S6.1.3 TCR

We examined a dataset of 22,004 TCRβ sequences measured in Ramien et al. (2019), taken from CD8+ T cells from a single healthy control patient (number HC12 in the study) in the 3rd trimester of pregnancy. We trained a ICA-MuE model as described above (Section S6.1.2), with 5 latent dimensions and a latent sequence length of 170. We used stochastic variational inference, with a learning rate of 0.01 and batch size of 5 over 2 epochs, annealing the prior KL divergence linearly over 1 epoch. Using 20% of the data as heldout validation, the model achieves a per residue perplexity of 2.39 on both the training and test datasets. We sampled from the model using the same strategy as in Section S6.1.2. The reference sequence used to construct $\hat{\psi}_{\text{ref}}$ was a randomly selected sequence from the dataset described in Section S6.6.2, which Tcellmatch predicted to bind the influenza epitope; as with the GFP example, conditioning on $\hat{\psi}_{\text{ref}}$ is a very weak form of supervision, learning from only a single functional example. In particular, the reference sequence was $MSNQVLCCVVLCLLGANTVDGGITQSPKYLFRKEGQNVTLSCEQNLNHDAMYWYRQ$ $DPGQGLRLIYYSQIVNDFQKGDIAEGYSVSREKKESFPLTVTSAQKNPTAFYLCASSIRSAYEQ$ $YFGPGTRLTVTEDLKNVFPPEVAVFEPSE$. The generated sequences had length $L = 149$.

S6.2 Synthesis Model Hyperparameters

In this section we describe the details of our stochastic synthesis models and optimization procedure. We used $K = 5$ pools, with L_k of approximately the same length for each $k \in \{1,\ldots,K\}$ (the last template was shortened as necessary since L is not always a multiple of 5). This yields templates of length 29 to 48 amino acids across all the datasets considered, which is consistent with typical oligosynthesis lengths of ~ 150 nucleotides. We used $A = 8$ for finite nucleotide mixtures; this value is realistic, as the company IDT, for example, currently offers four custom mixtures per oligo plus preset mixtures and single nucleotides (Pazdernik and Bowersox, 2016). We used $A = 24$ for finite codon mixtures, which is similar to typical trimer-based synthesis projects, which use the 20 amino acids plus a few custom mixtures (McMahon et al., 2018).

We set the mutation matrix S based on the ePCR enzyme Mutazyme II, available as part of Agilent's GeneMorph II Random Mutagenesis Kit `https://www.chem-agilent.com/pdf/strata/200550.pdf`. In particular, we converted the reported mutational spectra (Table II) into a substitution matrix, under the assumption that the test sequences are 50% A-T base pairs and 50% G-C base pairs: for instance, the probability of a particular base pair mutating per round of mutagenesis is given as 1% overall (10 bases per kilobase), and 50.7% of mutations happen to A-T base pairs, so the probability of a particular A-T base pair mutating is $0.01 \cdot 0.507/0.5$.

Proceeding in this way, we find

$$S = \begin{pmatrix} 0.990 & 0.006 & 0.005 & 0.003 \\ 0.006 & 0.990 & 0.003 & 0.005 \\ 0.003 & 0.001 & 0.991 & 0.001 \\ 0.001 & 0.003 & 0.001 & 0.991 \end{pmatrix} \tag{S29}$$

where the columns and rows are each in the order A, T, G, C. We also computed a mutation matrix S based on the Taq error prone polymerase (also in Table II of the Gene Morph II Random Mutagenesis Kit manual), but preliminary experiments suggested worse performance than Mutazyme II at matching the DHFR Potts target distribution, so we did not pursue it further. We limit the total number of rounds of mutagenesis τ to be less than 10, since large numbers of mutagenesis rounds are rarely used in practice.

Note that since we have chosen $A \geq 4$, the set of allowed values of $\mathcal{U}$ for enzymatic mutagenesis (that is, for all τ) is a strict subset of the set of allowed values of $\mathcal{U}$ for finite nucleotide mixtures (that is, for all v); thus, synthesis models using enzymatic mutagenesis are strictly less expressive than those using finite nucleotide mixtures. Meanwhile, $\mathcal{U}$ for finite nucleotide or codon mixtures is a strict subset of $\mathcal{U}$ for arbitrary codon mixtures, regardless of the choice of v; so synthesis models using finite mixtures are strictly less expressive than those using arbitrary codon mixtures.

S6.3 Baseline Synthesis Model

As a baseline stochastic synthesis approach, we considered a method motivated by a common heuristic for producing diversified libraries, which is to simply perform error prone PCR on an initial set of sequences. In particular, the baseline approach we consider is to do MC synthesis plus enzymatic mutagenesis: sample initial protein sequences $X'_1, \ldots, X'_M \sim p$, inverse-translate the protein sequences into DNA (sampling uniformly among all codons for the same amino acid), synthesize the DNA individually, and then mutagenize in the laboratory using ePCR. The distribution of resulting sequences can be described using a stochastic synthesis model for which we do *not* optimize the parameters. In particular, let $K = 1$, and for $m \in \{1, \ldots, M\}$ and $j \in \{1, \ldots, L\}$, let $\chi'_{mj1b} = 1$ if the sampled codon for X'_{mj} has base b at the first position, and $\chi_{mj1b} = 0$ otherwise. Likewise for χ_{mj2b} and χ_{mj3b}. Then, we set $u_{1mj} = S^\tau \chi_{mj1} \otimes S^\tau \chi_{mj2} \otimes S^\tau \chi_{mj3}$. We use fixed assembly, setting $w = (\frac{1}{M}, \ldots, \frac{1}{M})$. Then, the complete synthesis model (Equation 1) describes the distribution of sequences produced by the baseline approach.

Note that the baseline is effectively a kernel density estimate of p. It is thus unsurprising that the baseline underperforms relative to variational synthesis, since kernel density estimates typically underperform compared to mixture models.

Practically, we use S corresponding to a Mutazyme II enzyme (Section S6.2) and set $\tau = 5$ as a typical value for proteins of the length considered here (Wilson and Keefe, 2001). The samples from p used as initial sequences, $X'_1, \ldots, X'_M$, are subsampled from the same training dataset of 100,000 sequences used for variational synthesis (Section S6.4). We examined the performance of the method averaged over 3 independent sets of initial sequences.

S6.4 Optimization and Perplexity Evaluation

DHFR Potts, GFP, TCR To optimize synthesis models, we drew $\tilde{N} = 100,000$ samples from each target distribution p and applied stochastic EM, as described in Section S3.2. We chose batch sizes to be as large as possible without running out of memory. In particular, we used batch sizes of 100,000 (the full dataset) with $M = 1$, $M = 10$ and $M = 100$, and batch sizes of 10,000 for $M = 1000$. We trained for 80 epochs with $M = 1$, $M = 10$ and $M = 100$, and 16 epochs for $M = 1000$. Training took 2-5 minutes for each target-synthesis pair using a Tesla V100 GPU. Example training curves are shown in Figure S1.

Each synthesis model was trained on the same set of $\tilde{N} = 100,000$ samples from each target distribution, and evaluated based on the average per residue perplexity on the training dataset, $\exp(-\frac{1}{\tilde{N}} \sum_{i=1}^{\tilde{N}} \frac{1}{\ell(X_i)} \log q_\theta(X_i))$, where $\ell(X_i)$ is the length of the sequence X_i. Note that we do not perform heldout evaluation, as our goal is to see how well each synthesis model can match a target library of size 100,000; overfitting the synthesis model is not a concern, and may even help downstream performance, as described in Section S3.3. We initialized each optimization from three random seeds, and chose the result with the lowest perplexity.

DHFR raw For the DHFR raw dataset, we handle variable length sequences as described in Section S3.4, and

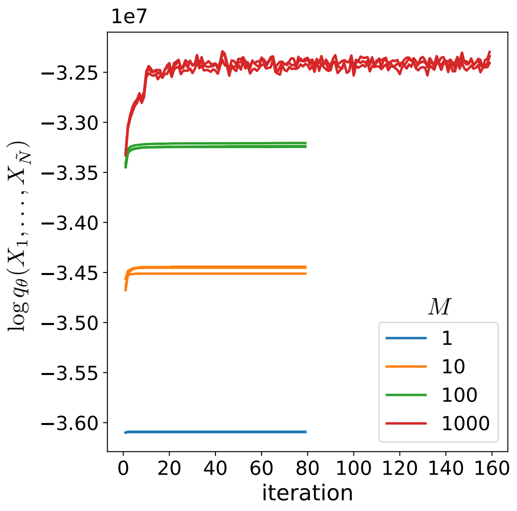

Figure S1: Illustrative example of training curves for a stochastic synthesis model (enzymatic mutagenesis with fixed assembly) with different values of M. For each value of M, training is repeated with three initial seeds. The models are each trained on samples from the DHFR Potts model, as described in Section S6.4.

optimized each synthesis model using EM with batch size of 3,629 (the full dataset), for 100 epochs. We set L to be the maximum length of sequences in the dataset including the stop codon, 170. We evaluated using mean per residue perplexity on the full dataset. We initialized each model from three random seeds, and chose the result with the lowest perplexity.

S6.5 BEAR Two-Sample Test

We use the vanilla version of the BEAR two-sample test proposed in Section 5 of Amin et al. (2021) to compare the target and the synthesis distributions. The test computes the Bayes factor comparing the hypothesis that two datasets $\{X_1, \ldots, X_{\tilde{N}}\}$ and $\{X'_1, \ldots, X'_{\tilde{N}'}\}$ come from the same underlying distribution versus different distributions. It uses $p_{\text{BEAR}}(X_1, \ldots, X_{\tilde{N}}|\alpha, \lambda)$, the probability of the dataset under a Bayesian Markov model with Dirichlet concentration parameter α and lag λ. In particular, the Bayes factor is

$$\text{BF} = \frac{p_{\text{BEAR}}(X_1, \ldots, X_{\tilde{N}}, X'_1, \ldots, X'_{\tilde{N}'})}{p_{\text{BEAR}}(X_1, \ldots, X_{\tilde{N}})p_{\text{BEAR}}(X'_1, \ldots, X'_{\tilde{N}'})} \tag{S30}$$

where

$$p_{\text{BEAR}}(X_1, \ldots, X_{\tilde{N}}) = \frac{1}{\Lambda}\sum_{\lambda=1}^{\Lambda} p_{\text{BEAR}}(X_1, \ldots, X_{\tilde{N}}|\alpha, \lambda). \tag{S31}$$

We used the training dataset of 100,000 samples from p as the first dataset in the two-sample test, and 100,000 independent samples drawn from the optimized synthesis model $q_{\hat{\theta}_{\tilde{N}}}$ as the second dataset. (In the case of DHFR raw, we used the 3,629 sequences as the target sample.) Note that the goal here is to understand whether the particular set of $\tilde{N}$ samples from p used for training look like a plausible set of samples from $q_{\hat{\theta}_{\tilde{N}}}$, following the logic of Section S3.3, so we do not resample from p to compute the test. We use $\alpha = 0.5$ and $\Lambda = 8$; we found that in general the posterior over lags concentrated at values of λ below 8, suggesting the test has sufficiently high resolution. Computing the test took about 5-10 minutes for each target-synthesis pair, with 20 cores on an Intel Xeon E5 v3 CPU.

S6.6 Sequence-to-Function Predictors

S6.6.1 GFP: TAPE

We computed TAPE predictions of GFP fluorescence using the interface in the FLEXS package (Sinai et al., 2020). Sequences with internal stop codons were assigned the minimum log fluorescence in the Sarkisyan et al.

(2016) dataset, 1.2. Variants with predicted log fluorescence above 3 were classified as hits, in line with the analysis of Sarkisyan et al. (2016) who classify variants below 3 as dark.

S6.6.2 TCR: Tcellmatch

We used Tcellmatch, trained on the same single-cell TCR sequencing data as in the original paper (10x Genomics, 2019), with the suggested model architecture (1x1 convolutional embeddings based on BLOSUM50 and biGRU layers). We used the mean squared logarithmic error to evaluate the model's ability to predict MHC multimer binding counts. We trained the Tcellmatch model only on TCRβ sequences, since the target p was trained only on TCRβ sequences. The Tcellmatch model uses only the CDR3 region to make predictions. In general, techniques for identifying the CDR3 region in TCRs rely on nucleotide-level information, which is unavailable for generated amino acid sequences. However, we constructed the target p by conditioning on a latent alignment, which in turn is based on a reference sequence with nucleotide-level information (Section S6.1.3). We thus use the positions corresponding to the CDR3 in the reference sequence (109:122, as annotated by the 10x pipeline) to define the CDR3 for each sampled sequence from p and q_θ. Although the Tcellmatch model can be used to predict many different antigens, we focused on predictions of the $GILGFVFTL$ influenza antigen, since the model had the most accurate predictions for this particular antigen (according to the R^2 metric used by Fischer et al. (2020), in particular $R^2 = 0.70$). We conditioned on a single donor (donor 1) when making predictions with Tcellmatch. Sequences with internal stop codons were assigned zero counts. Variants with predicted counts above 10 were classified as hits, in line with the analysis of Fischer et al. (2020).

S6.6.3 Estimating Hit Rates

Given a dataset of indicators for whether or not each of 100,000 samples from q_θ was a hit or not, i.e. $\{f(X_1), \ldots, f(X_N)\}$ where $f : \mathcal{X} \mapsto \{0, 1\}$, we estimated the overall hit rate using a Beta$(0.5, 0.5)$ prior (Jeffreys prior). We report the standard deviation of the posterior in Figure 4C and G.

S6.6.4 Estimating the Number of Unique Hits

Based on the hit rate (Section S6.6.3), we can estimate the total number of hits for libraries of any size. However, we are also interested in the total number of unique hits, since discovering identical sequences is not as useful as discovering diverse sequences. Evaluating predictors on very large numbers of samples, though, can be impractical since predictors (especially TAPE) can be computationally expensive. Instead, we used a Good-Toulmin estimation strategy: we examined the hits from a sample of 100,000 sequences from q_θ and then extrapolated to estimate the number of unique hits in a library of 1,000,000 sequences. We used the smoothed Good-Toulmin estimator proposed by Orlitsky et al. (2016), with the recommended Binomial model. Note that the estimator is considered trustworthy for datasets up to a factor of $\log N$ larger than the initial dataset; since $\log(10^5) = 11.5 \geq 10$, it is applicable here. We estimate the variance of the estimate under resampling using the jackknife, which can be efficiently computed for the smoothed Good-Toulmin estimator (Efron and Stein, 1981).

S6.7 Error Bars

In this section we summarize the calculation of the error bars in Figures 3 and 4. For the baseline model, we show the estimated standard deviation across independent samples of the initial M sequences from p (that is, the initial sequences that are mutagenized by ePCR). We use three independent samples for each value of M. For perplexity plots (Figures 3ACD and 4AE) we do not include any error estimates for non-baseline models, since we have exactly computed the total perplexity across the training dataset, and we are only interested in the match between the synthesis model and the training dataset, not in the synthesis model's generalization performance (as explained in Section S3.3). Bayes factors are themselves measurements of statistical significance, so we do not include any error bars for non-baseline models in Figures 3B and 4BF. For plots of hit rate (Figure 4CG), error bars show the posterior standard deviation of the hit rate under a Beta$(0.5, 0.5)$ prior (the Jeffreys prior) (Section S6.6.3). For plots of estimated unique hits (Figures 4DH), error bars show the jackknife estimate of the standard deviation (Section S6.6.4) For the baseline model, in plots of both hit rate and unique hits, error bars include the variance across different initial sequences from p, and are computed using the law of total variance.

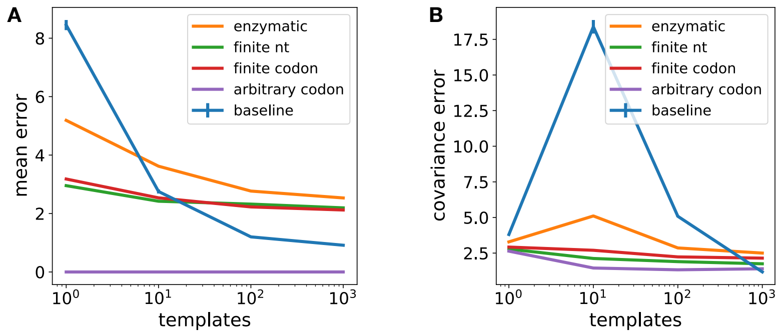

Figure S2: (A) Difference in mean between synthesis and target models, for various stochastic synthesis models with fixed assembly applied to the DHFR Potts target. Mathematically, $\|\mathbb{E}_{q_\theta}[X]-\mathbb{E}_p[X]\|_2$ where X is represented as a one-hot encoding and $\|\cdot\|_2$ is the Euclidean distance. (B) Difference in position-wise covariance matrices between synthesis and target models. Mathematically, let $\tilde{C}^{(p)}_{j,j',d,d'} := \text{Cov}_p(X_{j,d}, X_{j',d'})$ denote the covariance under p between the dth amino acid at position j and the d'th amino acid at position j'. The magnitude of the covariance between positions j and j' can be measured as $\mathcal{C}^{(p)} := \|\tilde{C}^{(p)}_{j,j'}\|_2$. Then we plot the position-wise covariance error $\|\mathcal{C}^{(p)} - \mathcal{C}^{(q_\theta)}\|_2$. In both plots, error bars for the baseline model are the standard deviation over initial sequences (Section S6.7).

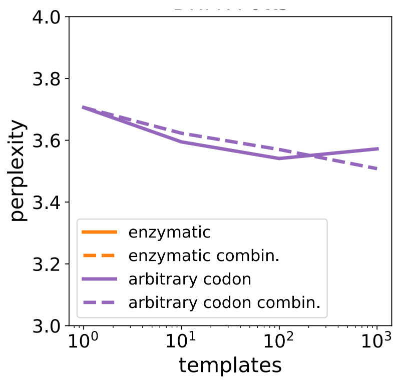

Figure S3: Zoom in of Figure 3C.

S6.8 Additional results

S6.8.1 DHFR

We further examined the match between stochastic synthesis models and the target DHFR Potts model, examining the difference in moments of each distribution. In particular, we looked at the difference in the mean sequence produced by the synthesis and target distributions, and the difference in covariance between positions of the sequences produced by the synthesis and target distributions (Figure S2). Comparing different variational synthesis models, we see improved perplexity (Figure 3A) corresponds well with lower moment error (Figure S2). Interestingly, the baseline synthesis method (Section S6.3) yields comparatively low moment error for large M despite comparatively poor perplexity.

We further examined the difference in performance between combinatorial and fixed assembly models. For enzymatic mutagenesis, switching from fixed to combinatorial assembly improves the two-sample test Bayes factor (Figure S4A), mean (Figure S4B) and covariance (Figure S4C). For arbitrary codon synthesis, switching from fixed to combinatorial assembly slightly improves the Bayes factor (Figure S4A), has no effect on the mean (as we expect mathematically and see in Figure S4B), but substantially worsens the covariance (Figure S4C). These results illustrate how the advantages of using fixed versus combinatorial assembly vary depending on the codon diversification technology.

We further examined the performance of different stochastic synthesis models applied to the DHFR raw dataset

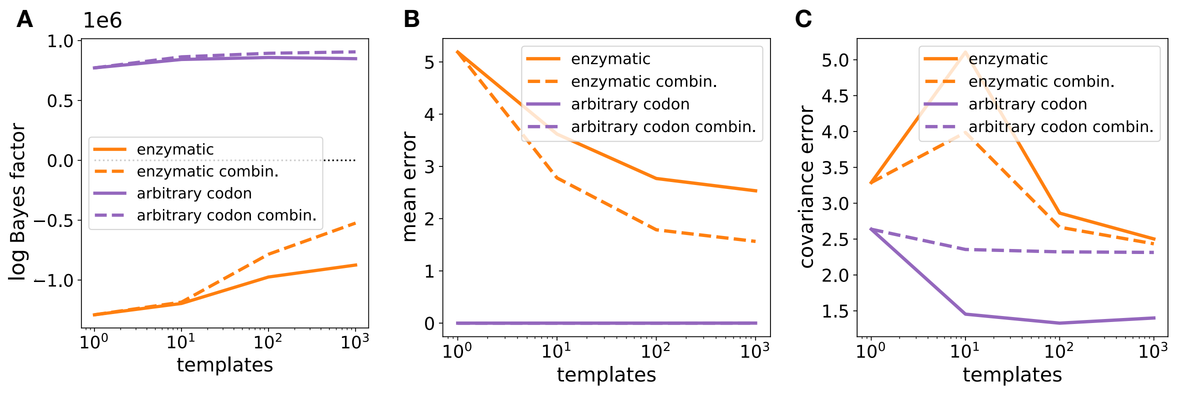

Figure S4: Comparing fixed versus combinatorial assembly for the DHFR Potts target. (The perplexity comparison can be found in Figure 3C and S3.) (A) Two-sample test Bayes factor. (B) Difference in mean between synthesis and target models, as defined in Figure S2. (C) Difference in position-wise covariance matrices between synthesis and target models, as defined in Figure S2.

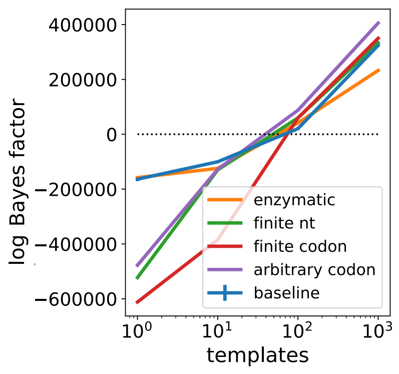

Figure S5: Two-sample test Bayes factor for synthesis models with fixed assembly applied to the DHFR raw dataset. For perplexity comparison, see Figure 3D.

of unaligned evolutionary sequences. Applying the two-sample test, we find that using large numbers of templates with any codon diversification technology is better than using small numbers of templates with a very expressive codon diversification technology (Figure S5), in line with the perplexity results (Figure 3D). We also see that variational synthesis is capable of matching the target closely enough to pass the two-sample test, but so is the baseline method in this case.

S6.8.2 GFP

We examined the difference in moments between the target GFP distribution and the stochastic synthesis models. The results (Figure S7) are qualitatively similar to those described for DHFR (Section S6.8.1 and Figure S2), with the baseline model performing better than its perplexity would suggest.

We examined the difference in average log fluorescence between samples from various stochastic synthesis models, as compared to exact samples from the target (that is, as compared to the average log fluorescence under MC synthesis) (Figure S8). Interestingly, we find that while using finite codon mixtures with $M = 1$ yields relatively low hit rates compared to arbitrary codon mixtures with $M = 1$ (Figure 4G), it yields nearly equivalent average log fluorescence (Figure S8).

We examined the difference in performance between combinatorial and fixed assembly methods applied to the GFP target distribution. On statistical measures of the difference between the synthesis and target distribution (Figure S9ABCD), we find broadly similar effects to those observed for DHFR Potts: for instance, we see moderate improvements in perplexity for enzymatic mutagenesis at large M when switching from fixed to combinatorial assembly, but little effect for arbitrary codon mixtures, and substantially worse covariance for arbitrary

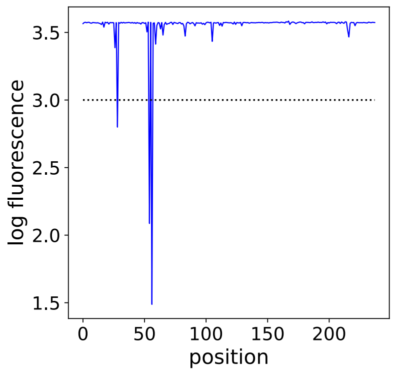

Figure S6: Predicted mutational effects of substituting each position of the stabilized wild-type GFP sequence with an alanine (i.e. an *in silico* alanine scan). Dotted line shows the threshold for classifying a variant as functional.

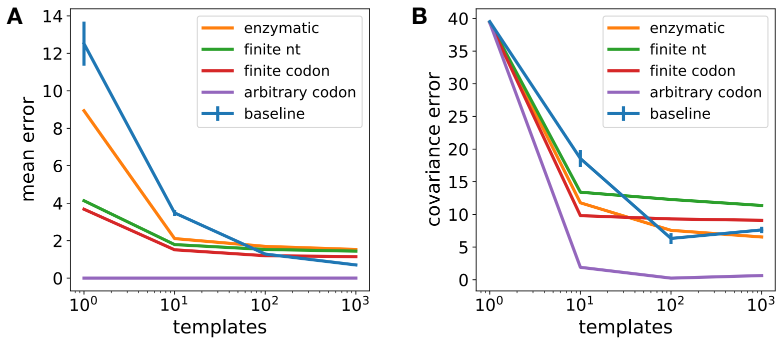

Figure S7: (A) Difference in mean between GFP synthesis and target models, as defined in the caption of Figure S2. (B) Difference in position-wise covariance matrices between GFP synthesis and target models, as defined in the caption of Figure S2.

codon mixtures. On measures of function, using combinatorial assembly leads to dramatically worse performance (Figure S9EFG): using arbitrary codon mixtures with combinatorial instead of fixed assembly drops the number of unique hits by three orders of magnitude. This result suggests that passing the BEAR two-sample test with large Bayes factors is not enough to ensure high hit rates when using combinatorial assembly; one should also inspect the covariance error.

S6.8.3 TCR

We examined the difference in moments between the target TCR distribution and the stochastic synthesis models. The results (Figure S11) are qualitatively similar to those described for DHFR (Section S6.8.1 and Figure S2), with the baseline model performing better than its perplexity would suggest.

We examined the difference in average binding counts between samples from various stochastic synthesis models, as compared to exact samples from the target TCR model (that is, as compared to MC synthesis) (Figure S12). Unlike for GFP, we find that the average value of the assay output is roughly proportional to the hit rate.

We examined additional viral epitopes, besides the influenza epitope, for which decent Tcellmatch predictions were available. The second highest quality Tcellmatch predictor ($R^2 = 0.43$) was for an Epstein-Barr virus (EBV) epitope, $RAKFKQLL$. MC synthesis with $N_0 = 10^3$ generates just 0.05 hits on average across independent libraries, while variational synthesis with $N_1 = 10^6$, using arbitrary codon mixtures and $M = 10$, generates an expected 30 unique hits (Figure S13). Here, variational synthesis makes the difference between likely failure and likely success. We also examined two viral epitopes for which the target TCR model had an estimated hit rate of zero (based on a sample of 10^5 sequences from the model): a cytomegalovirus epitope ($KLGGALQAK$, Tcellmatch $R^2 = 0.40$) and another EBV epitope ($RLRAEAQVK$, Tcellmatch $R^2 = 0.36$). Note that a hit rate

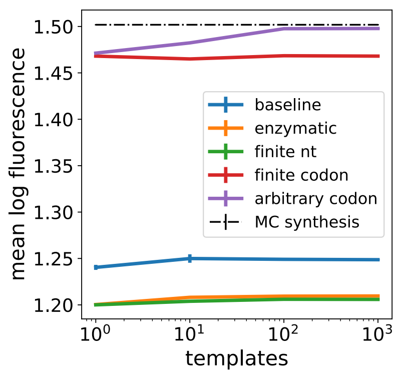

Figure S8: Average log predicted fluorescence of samples from various stochastic synthesis models and from the GFP target p itself (MC synthesis). Error bars are estimates of the standard deviation of the mean (for the baseline model, this includes variance across different initial sequences, as described in Section S6.7; for the rest of the models, it is just the standard error, and negligible in these plots).

of close to zero is unsurprising, given that the Tcellmatch predictor has low accuracy, and that the individual patient which the TCR model was trained on may not have TCRs that bind these epitopes. For these two epitopes, we found that variational synthesis was still able to closely match the average binding counts under the target TCR model (Figure S14).

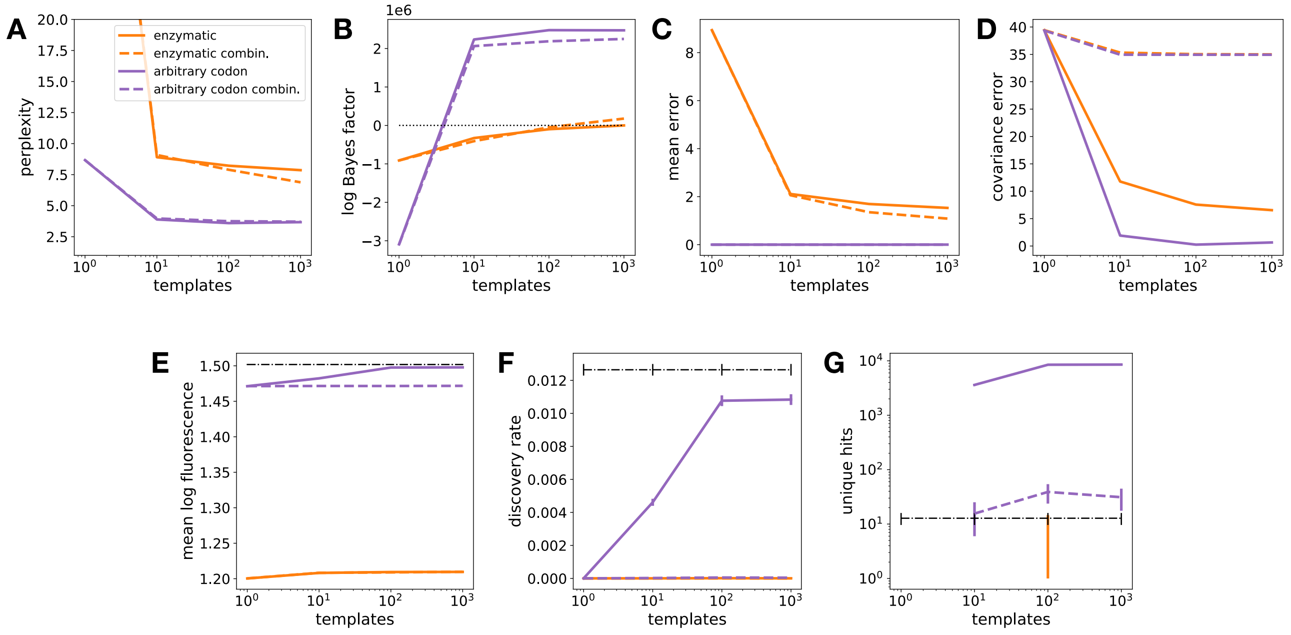

Figure S9: Fixed versus combinatorial stochastic synthesis applied to the GFP target distribution. (A) perplexity, (B) two-sample test Bayes factor, (C) mean error (as defined in Figure S2), (D) covariance error (as defined in Figure S2), (E) average log fluorescence, (F) hit rate and (G) number of unique hits with $N_1 = 10^6$ and $N_0 = 10^3$. Error bars are as described in Section S6.7.

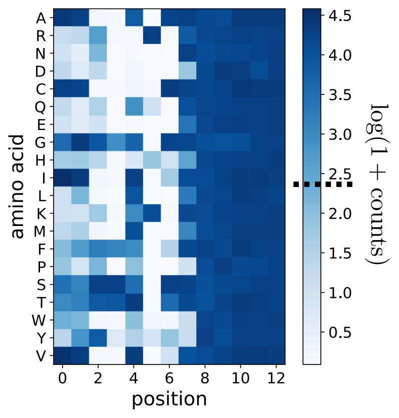

Figure S10: Predicted binding effects of substituting each position of a natural CDR3 sequence ($CASSIRSAYEQYF$) with each of 20 amino acids (*in silico* deep mutational scan). The threshold for functionality (10 counts) is marked by a dotted line in the colorbar.

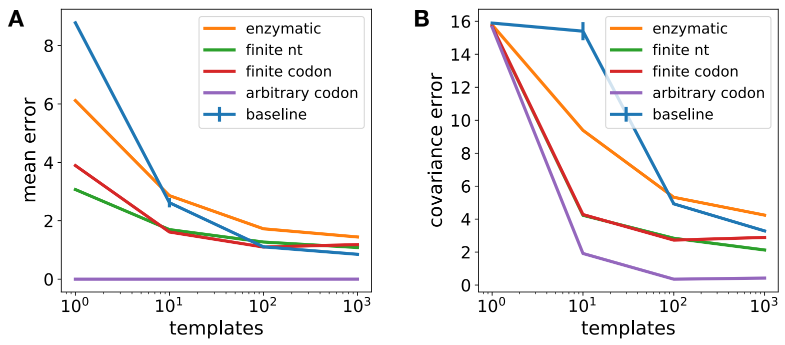

Figure S11: (A) Difference in mean between TCR synthesis and target models, as defined in the caption of Figure S2. (B) Difference in position-wise covariance matrices between TCR synthesis and target models, as defined in the caption of Figure S2.

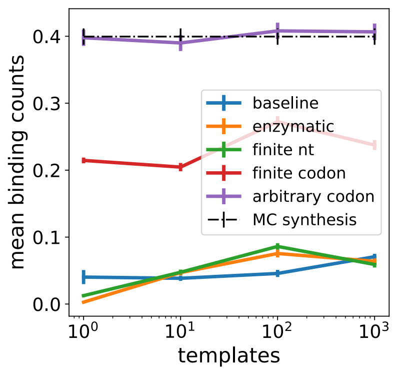

Figure S12: Average predicted binding counts of samples from various stochastic synthesis models and from p itself (MC synthesis). Error bars are estimates of the standard deviation of the mean (for the baseline model, this includes variance across different initial sequences; for the rest of the models, it is just the standard error).

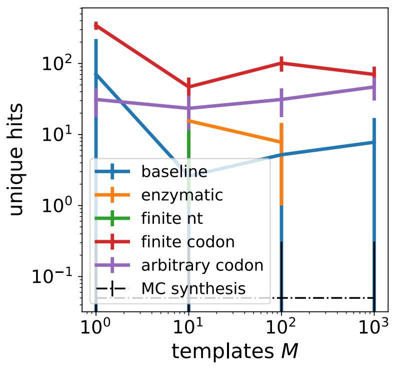

Figure S13: Same as Figure 4H, but for the EBV epitope $RAKFKQLL$ instead of the influenza epitope.

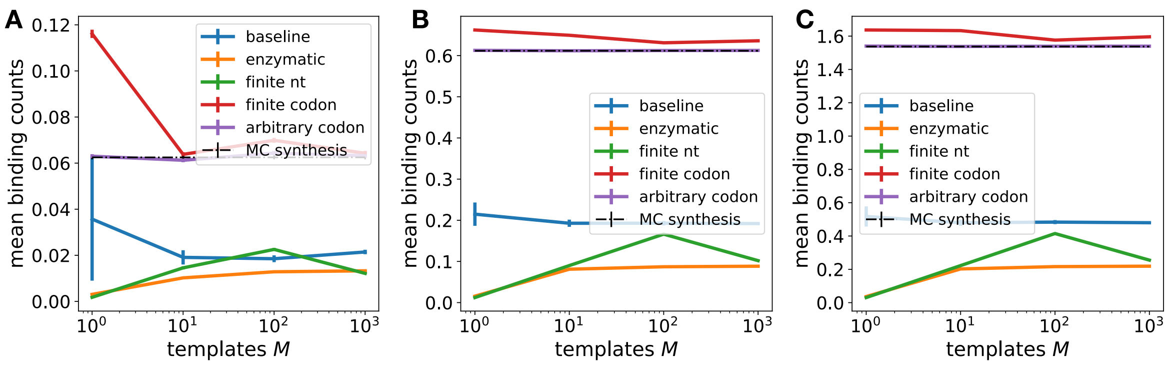

Figure S14: Same as Figure S12, but for the EBV epitope $RAKFKQLL$ (A), the EBV epitope $RLRAEAQVK$ (B) and the CMV epitope $KLGGALQAK$ (C).