A Intractability of factorizable library design

Let $f(.)$ denote the objective function we wish to optimize our library for, and suppose that $f(.)$ can be expressed as some multi-output boolean circuit of size at most $|\Sigma^L|$. The output bits could be interpreted as, for example, a floating point number.

To design a library without any constraints, we can simply enumerate and score all the sequences in time polynomial in $|\Sigma^L|$. However, once factorizability is enforced, then under the assumption that P $\neq$ NP there exists no algorithm that runs in time polynomial in $|\Sigma^L|$ that can guarantee a solution appreciably better than a random solution.

Theorem 3. *Let ϵ be any strictly positive constant that is at most 1. Let $f(.)$ be expressed as some multi-output boolean circuit of size at most $|\Sigma^L|$. Then unless $P = NP$, there exists no algorithm running in time polynomial in $|\Sigma^L|$ that can, upon receiving $f(.)$ as input and L_s, L_p, n_s, and n_p as parameters, find a factorizable library with factors $S_p \subseteq \Sigma^{L_p}$ and $S_s \subseteq \Sigma^{L_s}$ of sizes n_p and n_s respectively such that:*

$$(\sum_{s \in S_p \oplus S_s} f(s)) - \mu \geq \epsilon\Big(\big(\max_{S_1 \in P_1, S_2 \in P_2} \sum_{s \in S_1 \oplus S_2} f(s)\big) - \mu\Big) \tag{7}$$

Where $P_1 \subseteq 2^{\Sigma^{L_p}}$ are all the subsets of Σ^{L_p} of size n_p and $P_2 \subseteq 2^{\Sigma^{L_s}}$ are all the subsets of Σ^{L_s} of size n_s, and $\mu = \mathbb{E}[\sum_{s \in X \oplus Y} f(s)]$, where X and Y are uniformly random over P_1 and P_2.

Proof. We will prove the contraposition and show that P = NP is implied if such an algorithm exists.

If the algorithm described does exist, then there exists an algorithm A that can, upon being given $f : \Sigma^{2L} \to \{-1, 0, 1\}$ as a boolean circuit of size at most $|\Sigma^{2L}|$ with 2 output bits and $n \leq |\Sigma^L|$, return within time polynomial in $|\Sigma^L|$ a pair of sets $S_p \in \Sigma^L$ and $S_s \in \Sigma^L$ each of size n such that:

$$(\sum_{s \in S_p \oplus S_s} f(s)) - \mu_{f,n} \geq \epsilon\big(\phi_{f,n} - \mu_{f,n}\big) \tag{8}$$

Where $\mu_{f,n}$ denotes the expected score of a random solution and $\phi_{f,n}$ denotes the optimal solution. Let $P_n \subseteq 2^{\Sigma^L}$ be all subsets of Σ^L of size n, and let X and Y be drawn uniformly at random from P_n. Then formally:

$$\mu_{f,n} = \mathbb{E}[\sum_{s \in X \oplus Y} f(s)] \tag{9}$$

$$\phi_{f,n} = \max_{S_1 \in P_n, S_2 \in P_n} \sum_{s \in S_1 \oplus S_2} f(s) \tag{10}$$

Let $G = (V, E)$ be some graph that we wish to approximate the max clique of. Set $L = \lceil log_{|\Sigma|}(|V|^4) \rceil$, so $|V|^4 \leq |\Sigma^L|$. We then pick a subset of $V' \subseteq \Sigma^L$ to represent the vertices V. We can then construct a function $f : \Sigma^{2L} \to \{-1, 0, 1\}$:

$$f(x \oplus y) = \begin{cases} 1, & \text{if } x = y \text{ and } x \in V' \\ -1, & \text{if } x \text{ is not a neighbour of } y \text{ and } x, y \in V' \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

We can express f as a circuit that is polynomial in the size of G by constructing it to only care about sequences with prefixes and suffixes in V' and ignoring all other sequences and outputting 0. This will also ensure it is smaller than $|\Sigma^{2L}|$ (for Σ^{2L} sufficiently large), and can be done in time polynomial with respect to the size of G. We then feed f to algorithm A along with $n = |V|$. Let X be the score of the solution found by A.

Lemma 1. *Suppose $n \leq |V'|$. Then $\mu_{f,n} \geq -|V'|^{-2}$.*

Proof. First we note that for any $S_1 \subseteq \Sigma^L$ and $S_2 \subseteq \Sigma^L$, we have $\sum_{s \in S_1 \oplus S_2} f(s) \in [-|V'|^2, |V'|]$. This is because at most $|V'|$ sequences evaluate to 1 and at most $|V'|^2$ sequences evaluate to -1.

Next, consider the probability of drawing a random subset of Σ^L of size n that contains at least one sequence in V'. The probability of this is upper bounded by the expected number of sequences in V' in such a subset, which is $\frac{n|V'|}{|\Sigma^L|} \leq |V'|^{-2}$ by construction. Therefore, the probability of randomly drawing two subsets $S_1, S_2 \subseteq \Sigma^L$ of size n such that $\sum_{s \in S_1 \oplus S_2} f(s) \neq 0$ is at most $|V'|^{-4}$.

Therefore, $\mu_{f,n} \geq -\Pr(\sum_{s \in S_1 \oplus S_2} f(s) \neq 0)|V'|^2 \geq -|V'|^{-2}$, where S_1 and S_2 are uniformly random subsets of Σ^L of size n, as desired. □

Lemma 2. *Let $n = |V'|$. Then $\phi_{f,n}$ is equal to the size of the max clique in G.*

Proof. The only way a we can contribute positively to the score is if an element v belonging to V' is contained in both libraries. However, this contribution is negated if there is some sequence in V' contained in either library that is not a neighbour of v. Therefore, the best score can be no larger than the size of the max clique in G. Conversely, we can attain the size of the max clique as a score if we include the elements in V' that form the max clique in both libraries, and make every other sequence elements outside of V'. □

Then we have the following:

$$X - \mu_{f,n} \geq \epsilon(\phi_{f,n} - \mu_{f,n}) \tag{12}$$
$$X \geq \epsilon\phi_{f,n} + \mu_{f,n}(1-\epsilon) \tag{13}$$
$$X \geq \epsilon\phi_{f,n} - |V'|^{-2}(1-\epsilon) \tag{14}$$

For sufficiently large graphs, $|V'|^{-2}(1-\epsilon) \leq \frac{\epsilon}{2}\phi_{f,n}$ since $\phi_{f,n} \geq 1$ (we can include a single sequence of V' in both libraries and make sure every other sequence is not in V'). Therefore:

$$X \geq \epsilon\phi_{f,n} - \frac{\epsilon}{2}\phi_{f,n} \tag{15}$$
$$X \geq \frac{\epsilon}{2}\phi_{f,n} \tag{16}$$

Therefore, X is at least a $\frac{\epsilon}{2}$ fraction of the size of the max clique. Thus, we have constructed a procedure for attaining a $\frac{\epsilon}{2}$ approximation of the size of the maximal clique that runs in time polynomial in $|\Sigma^{2L}|$ (via the properties of algorithm A), which scales polynomially with the size of the graph. Since any constant factor approximation of the size of the maximum clique is known to be NP-hard [Zuckerman, 2006], we attain P = NP as desired. □

B Proof of statements from the main text

B.1 Proof of Theorem 1

Let $A_{x,i}$ denote the ith entry in the vector $\varphi_p(x)$. Let $B_{y,i}$ denote the ith entry in the vector $\varphi_s(y)$. Let $F_{x,y}$ denote $f(x \oplus y)$. If we assign a total ordering to elements of the prefix and suffix spaces and treat elements of those spaces as the numbers denoting their ordering, then we can view A, B, and F as matrices. We can then rewrite the equation from the statement of Theorem 1:

$$\frac{\sum_{x \in \Sigma^{L_p}} \sum_{y \in \Sigma^{L_s}} \left(f(x \oplus y) - \varphi_p(x) \cdot \varphi_s(y)\right)^2}{\sum_{s \in \Sigma^L} f(s)^2} \leq 1 - \frac{m}{|\Sigma^{\min(L_p, L_s)}|} \tag{17}$$

as the following:

$$\frac{\|F - AB^T\|^2}{\|F\|^2} \leq 1 - \frac{m}{|\Sigma^{\min(L_p, L_s)}|} \tag{18}$$

Where $\|.\|$ denotes the Frobenius norm of a matrix. By definition, AB^T is a rank m' matrix. Let $USV = F$ be the singular value decomposition of F. By the Matrix Approximation Lemma, $\|F - AB^T\|$ is minimized when $AB^T = U\tilde{S}V$, where $\tilde{S}$ is S where only the largest m' values along the diagonal are not zero (if $m' \geq |\Sigma^{\min(L_p,L_s)}|$, then A and B can always be chosen such that $AB^T = F$, so $\|F - AB^T\| = 0$). Therefore:

$$\min_{A,B}\left(\frac{\|F - AB^T\|^2}{\|F\|^2}\right) = \frac{\|U(S - \tilde{S})V\|^2}{\|USV\|^2} = \frac{tr((S - \tilde{S})^2)}{tr(S^2)} \leq \frac{|\Sigma^{\min(L_p,L_s)}| - \min(m', |\Sigma^{\min(L_p,L_s)}|)}{|\Sigma^{\min(L_p,L_s)}|} \tag{19}$$

Therefore, Equation 18 holds for all F if $m' \geq m$. If $m' < m$, then we note that the last inequality in Equation 19 holds with equality if S is a matrix with zero entries everywhere except along the main diagonal, which implies that there is some F for which Equation 18 does not hold for any A and B. Theorem 1 follows.

B.2 Proof of Theorem 2

Let $p_{i,c}$ denote the fraction of sequences with character c at position i. We can then write $\mathcal{H}(S_p, S_s)$ as the following:

$$\mathcal{H}(S_p, S_s) = -|S_p \oplus S_s| \sum_{i=1}^{L} \sum_{c \in \Sigma} p_{i,c} ln(p_{i,c}) \tag{20}$$

We also have the following relations:

$$\forall i \sum_{c \in \Sigma} p_{i,c} = 1 \tag{21}$$

$$\sum_{i=1}^{L} p_{i,d_i} = R \tag{22}$$

Where R is some constant that is at least $L - m$ and d_i denotes the ith character of d. The first relation follows since the proportions must add up to 1, and the second relation follows since all sequences can be obtained with m substitutions of d.

$\mathcal{H}(S_p, S_s)$ is upper bounded by the maximum value $-|S_p \oplus S_s| \sum_{i=1}^{L} \sum_{c \in \Sigma} p_{i,c} ln(p_{i,c})$ can take subject to the constraints imposed by Equations 21 and 22. The optimization problem can be solved via Lagrange multipliers, which yields the following critical point:

$$\forall i \, \forall c \neq d_i \; p_{i,c} = \frac{1 - R/L}{|\Sigma| - 1} \tag{23}$$

$$p_{i,d_i} = \frac{R}{L} \tag{24}$$

Since this is the only critical point, and since it is straightforward to find $p_{i,c}$ under the constraints such that the objective evaluates to a lower value (set p_{i,c_i} to $1-R/L+\epsilon$ for all i where $c_i \neq d_i$ is some arbitrarily chosen symbol and where ϵ approaches arbitrarily close to 0), this must be a global maximum. Therefore:

$$\begin{aligned}
\mathcal{H}(S_p, S_s) &\leq -|S_p \oplus S_s| \sum_{i=1}^{L} \big(\frac{R}{L} ln(\frac{R}{L}) + \sum_{i=1}^{|\Sigma|-1} \frac{1-R/L}{|\Sigma|-1} ln(\frac{1-R/L}{|\Sigma|-1})\big) && (25)\\
&\leq -|S_p \oplus S_s| L\big((\frac{R}{L}) ln(\frac{R}{L}) + (1-\frac{R}{L}) ln(1-\frac{R}{L}) + (1-\frac{R}{L}) \sum_{i=1}^{|\Sigma|-1} \frac{1}{|\Sigma|-1} ln(\frac{1}{|\Sigma|-1})\big) && (26)\\
&\leq |S_p \oplus S_s| L\big(ln(2) + (1-\frac{R}{L}) ln(|\Sigma - 1|)\big) && (27)\\
&< |S_p \oplus S_s| L\big(ln(2) + (1-\frac{R}{L}) ln(|\Sigma|)\big) && (28)\\
&\leq |S_p \oplus S_s| L\big(ln(2) + \frac{m}{L} ln(|\Sigma|)\big) && (29)
\end{aligned}$$

The final inequality follows from $R \geq L - m$. Dividing the overall inequality by $L|S_p \oplus S_s| ln(|\Sigma|)$ yields the desired result.

C Additional details for methods

C.1 Small feature spaces for Potts models

If $f(x)$ gives the energy of $x \in \Sigma^L$ and can be described by a Potts model, then a pair of feature maps $\varphi_p(.) : \Sigma^{L_p} \to \mathbb{R}^{\min(L_p,L_s)|\Sigma|+2}$ and $\varphi_s(.)\Sigma^{L_s} \to \mathbb{R}^{\min(L_p,L_s)|\Sigma|+2}$ can be found. First, we express sequences $x \in \Sigma^L$ as a tensor where $x_{i,c} = 1$ if $x_i = c$ and 0 otherwise. Then there must exist some rank 4 tensor A such that the following holds:

$$\begin{aligned}
f(x) =& \sum_{i=1}^{L} \sum_{c_i \in \Sigma} \sum_{j=1}^{L} \sum_{c_j \in \Sigma} x_{i,c_i} x_{j,c_j} A_{i,c_i,j,c_j} && (30)\\
=& \sum_{i=1}^{L_p} \sum_{c_i \in \Sigma} \sum_{j=1}^{L_p} \sum_{c_j \in \Sigma} x_{i,c_i} x_{j,c_j} A_{i,c_i,j,c_j} + \sum_{i=L_p+1}^{L} \sum_{c_i \in \Sigma} \sum_{j=L_p+1}^{L} \sum_{c_j \in \Sigma} x_{i,c_i} x_{j,c_j} A_{i,c_i,j,c_j} && (31)\\
&+ \sum_{i=1}^{L_p} \sum_{c_i \in \Sigma} \sum_{j=L_p+1}^{L} \sum_{c_j \in \Sigma} (x_{i,c_i} x_{j,c_j} A_{i,c_i,j,c_j} + x_{i,c_i} x_{j,c_j} A_{j,c_j,i,c_i}) && (32)\\
=& g(x') + h(x'') + \sum_{i=1}^{L_p} \sum_{c_i \in \Sigma} \sum_{j=1}^{L_s} \sum_{c_j \in \Sigma} x'_{i,c_i} x''_{j,c_j} (A_{i,c_i,j+L_p,c_j} + A_{j+L_p,c_j,i,c_i}) && (33)\\
=& g(x') + h(x'') + \tilde{x}'^T B \tilde{x}'' && (34)
\end{aligned}$$

Where x' is the matrix representing the prefix of x and x'' is the matrix representing the suffix of x. $\tilde{x}'$ and $\tilde{x}''$ are the flattened vectors of x' and x'', and B is some $L_p|\Sigma|$-by-$L_s|\Sigma|$ matrix obtained by reshaping and summing the appropriate subtensors of A.

Without loss of generality, let $L_p \leq L_s$. Then we can simply let $\varphi_p(x') = \tilde{x}' \oplus [1, g(x')]$ and $\varphi_s(x'') = B\tilde{x}'' \oplus [h(x''), 1]$, where here we use $\oplus$ to denote vector concatenation. Therefore both prefix and suffix feature maps map to $\mathbb{R}^{\min(L_p,L_s)|\Sigma|+2}$, and $\varphi_p(x') \cdot \varphi_s(x'') = f(x)$ as desired.

C.2 The objective function scales with library size

Suppose we propose to change a sequence x to x' in the prefix library, where x and x' differ by only a single residue at position i. Let l be the length of x and x'. Let S'_p denote the updated prefix library. The difference in objective between the initial library and the proposed library is then:

$$\mathcal{F}(S'_p, S_s) - \mathcal{F}(S_p, S_s) = \left(\sum_{s_s \in S_s} f(x' \oplus s_s) - f(x \oplus s_s) \right) + \lambda |S_s||S_p[l]| \Big(h(S'_p[l], i) - h(S_p[l], i) \Big) \quad (35)$$

The first term has a sum that contains $|S_s|$ terms. If our proposal is an improvement to most sequences it affects, then the size of the sum roughly scales with $|S_s|$. For the second term, if the library is sufficiently large we can estimate the change in entropy with the derivative evaluated at $t = 0$:

$$h(S'_p[l], i) - h(S_p[l], i) \approx \frac{d}{dt}(p_1 + \frac{t}{|S'_p[l]|}) ln(p_1 + \frac{t}{|S'_p[l]|}) - (p_2 - \frac{t}{|S'_p[l]|}) ln(p_2 - \frac{t}{|S'_p[l]|}) = \frac{1}{|S'_p[l]|} ln(\frac{p_1}{p_2}) \quad (36)$$

Where p_1 is the fraction of sequences in $S_p[l]$ that has x_i at position i and p_2 is the fraction of sequences in $S_p[l]$ that has x'_i at position i. Thus the second term in equation 35 also scales with $|S_s|$.

So the overall difference in score also scales with $|S_s|$. Similarly, we can see that making a change to a sequence in the suffix library induces a change that scales with $|S_p|$. Thus in order to maintain a stochastic phase in the simulated annealing procedure we normalize the score as described in the main text.

C.3 Deep learning architecture

We employ a convolutional neural network with residual connections to map strings to real valued vectors. First, the strings are encoded into a 40-by-n array, where n is the maximum size of the string that the model can take as input. This is done by first padding out the string to maximum length, and taking each character in the string and mapping them into a 40 dimensional embedding, where the first 20 entries consists of a one-hot encoding denoting the residue and the last 20 entries is the BLOSUM62 substitution values [Eddy, 2004]. The padding character is assigned the zero vector. These vectors then make up the columns of the resulting encoded matrix and is fed into the neural network, where convolutions are run over the second dimension and the first dimension is treated as a channel dimension.

The model first applies a linear transform to each position that increases the number of channels from 40 to 64, followed by batch normalization. This is then run through 5 residual blocks. Each residual block consists of a pair of 1D convolutions with a kernel of size 3 and batch normalizations, separated by a ReLU layer in between. The input of the block is then added to the output, forming the residual block. Max pooling over adjacent positions is performed after each of the last 2 residual blocks. The resulting matrix is then flattened and linearly mapped to a 128 dimensional vector, which is then passed through a ReLU layer, which is then linearly mapped to an output vector.

For this work we make use of three kinds of models. The unrestricted model accepts length 20 sequences and outputs 1 dimensional vectors, which are treated as output scores. The independent model consists of a prefix and a suffix model that accept length 10 sequences and output 1 dimensional vectors, which are added together to give the final output scores. The sequence is first padded to length 20, and then divided into two sequences of length 10 which are fed into the prefix and suffix models. The reverse kernel model operates identically to the independent model, except the models output 16 dimensional vectors. The dot product of the vectors then give the final score.

C.4 Training deep learning models

Training datasets were split into 10 equal sets (one set has slightly more or less sequences when the number of sequences is not a multiple of 10). These were used to create 10 training and validation splits, where for each split one of the sets was used for validation and the rest were combined for training. For each split, two randomly intialized models were trained, resulting in an ensemble of 20 models. Final predictions are made by averaging over the outputs of the ensemble. Each model was trained for 100 epochs using ADAM with default PyTorch v1.7 parameters [Kingma and Ba, 2014]. Model performances were evaluated using the validation set after each epoch, and the model with the highest performance was saved. Models only accept sequences of length 20, so shorter sequences were randomly padded such that each shift occurs with equal probability. This allows the model to learn shift invariance, which is necessary for sequence generation since different shifts may be represented once prefix and suffix libraries are concatenated. Models were created using PyTorch v1.7 and trained on either a single NVIDIA Titan RTX GPU (24GB RAM) or a single GeForce GTX 1080 Ti (11GB RAM). Training each model took around 30-60 minutes, although the best performing model was usually found within 15-30 minutes.

Held out validation sets were used to evaluate performance in Figure 3. These sets were filtered to exclude any sequences that overlapped the training sets. Sequences were padded equally on both sides during this evaluation, with extra padding applied to the end if the sequence length is odd.

D Additional details on benchmarking SAPS

D.1 Additional details on the problem domain

To test SAPS, we generate energy landscapes over the domain of fixed length binary strings that are defined by non-lattice Ising models. The Hamiltonian takes the following form:

$$H(s) = \sum_{i=1}^{n} \sum_{j=i}^{n} \sum_{c_i=\{0,1\}} \sum_{c_j=\{0,1\}} I_{i,j,c_i,c_j} \mathbb{1}_{s_i=c_i} \mathbb{1}_{s_j=c_j} \tag{37}$$

Where n is the length of the strings, s denotes a string of length n, and $\mathbb{1}_{x=y}$ is the indicator function that returns 1 when $x = y$ and 0 otherwise. To generate a single Ising model, I_{i,j,c_1,c_j} are all independently and uniformly drawn from $\{0, -1, 1\}$. 100 such models were generated for domains over sequences of length 14, 16, 18, and 20 for a total of 400 models.

This model permits us to define a a pair of practical feature maps as described in Appendix C.1 that allow us to efficiently use the reverse kernel trick.

Such a model can be seen as a toy model for protein design. For example, suppose we know the exact desired locations and orientations of the alpha carbons of the protein. That geometry then determines the interactions, which can then be captured with something akin to the model described above.

D.2 Alternative optimization approaches to SAPS

The first approach we benchmark against is the greedy approach, where we start with a randomly generated pair of libraries. We then sweep over each bit in each sequence in each library, flipping it if it produces an improvement. We keep sweeping until convergence. Note that this is equivalent to SAPS if the temperature parameter approaches zero.

For the next two approaches, we take each segment library and rank the sequences according to some heuristic. The two we use are the expectation and max heuristic. For the expectation heuristic, we take a segment and assign it the mean of all sequences containing that segment. For the max heuristic, we assign to the segment the optimal score of all sequences containing that segment. We then take the top scoring segments from each segment library.

Note that the runtime of each of these benchmarks (with the exception of the greedy approach) is $\Omega(|\Sigma^L|)$, where Σ^L is the set of all sequences. This is because it is necessary to score every single sequence in order to calculate the expectation and max heuristic. Therefore, the implementation of these heuristics in and of themselves is non-trivial and may require heuristics. However, our benchmarking landscapes are small enough that we can calculate them exactly. Thus, our benchmarks represent an optimistic view for how well these other approaches can perform.

In fact, if there are no interactions between the segments the max and expectation heuristics will give the optimal solution. Having no interactions means the Hamiltonian of the Ising model would take on the following form:

$$H(s) = \sum_{i=1}^{m} \sum_{j=i}^{m} \sum_{c_i=\{0,1\}} \sum_{c_j=\{0,1\}} I_{i,j,c_i,c_j} \mathbb{1}_{s_i=c_i} \mathbb{1}_{s_j=c_j} + \sum_{i=m+1}^{n} \sum_{j=i}^{n} \sum_{c_i=\{0,1\}} \sum_{c_j=\{0,1\}} I_{i,j,c_i,c_j} \mathbb{1}_{s_i=c_i} \mathbb{1}_{s_j=c_j} \tag{38}$$

Where the lengths of the strings is n, the lengths of the prefixes is m, and the lengths of the suffixes is $n - m$. We present the proof:

Proof. Let X and Y denote the set of prefixes and suffixes respectively. For any string $x \oplus y$ where $x \in X$ is a prefix and $y \in Y$ is a suffix, we have $H(x \oplus y) = H_p(x) + H_s(y)$ for some $H_p(.)$ and $H_s(.)$.

Let $E_p(.)$ denote the expectation heuristic on prefixes and let $M_p(.)$ denote the max heuristic on prefixes. Let $E_s(.)$ and $M_s(.)$ denote these heuristics on the suffixes. Let $x \in X$. We have $E_p(x) = H_p(x) + \mathbb{E}[H_s(Z)]$ and $M_p(x) = H_p(x) + \max_{z \in Y}(H_s(z))$, where Z denotes a random suffix distributed uniformly. Note that the second term is independent of x in both equations, therefore both heuristics would rank prefixes identically, and propose the prefixes with the highest $H_p(x)$. The identical argument shows that the heuristics would propose the suffixes with the highest $H_s(y)$.

Therefore, it suffices to show that the expectation heuristic provides the optimal solution. Suppose that there exists some pair of libraries that has a higher score than what's given by the heuristic. Then there must be some segment x that is in the better library and not in the heuristic library, and some segment y that is in the heuristic library but not in the better library such that $E_p(x) \leq E_p(y)$ (or $E_s(x) \leq E_s(y)$). Then swapping out x for y can not decrease the scores of any sequence, since $H_p(x) \leq H_p(y)$ (or $H_s(x) \leq H_s(y)$). If we keep swapping we will eventually obtain the library given by the heuristic. Since each swap cannot decrease the score, the heuristic library cannot score worse than the better library, which is a contradiction. □

For the final two approaches, we take the greedy approach, but instead of initializing with a random library we initialize with the outputs of the expectation and max heuristics. We refer to this as "greedy refinement" in the main text.

E Experimental details

E.1 Details of phage panning experiments and processing

The experimental single framework library (used in all panning experiments) was constructed as follows: a gene fragment encoding the germline framework combination IGHV3-23 and IGKV1-39 was synthesized in Fab format and cloned into a phagemid vector template. Only CDR-H3s were diversified via trinucleotide synthesis technology. Three of the four publicly available antibody-antibody targets were collected in-house and reported on in [Liu et al., 2020]. Ranibizumab is a Fab fragment that binds to vascular endothelial growth factor A, Etanercept is a fusion protein that fuses the TNF receptor 2 to the Fc and hinge region of a IgG1 heavy chain, and Trastuzumab is a monoclonal antibody that binds to human epidermal growth factor receptor 2 (HER2). The fourth, Omalizumab, is a IgG1k monoclonal antibody that specifically binds to free human immunoglobulin E. All were collected in the same fashion as follows: first, the targets were expressed in human IgG1 format. Three rounds of panning are completed in solid-phase mode on 96-well maxisorb plates that are coated with the target with decreasing concentrations over rounds. Finally, a negative control panning against no target for one round was conducted and is referred to as FW_kappa in the text. Panned phages are eluted and propagated for high throughput sequencing via MiSeq or HiSeq sequencers. For all experiments, after obtaining sequencing reads, the fixed flanking sequences on the boundary of the variable region were used as a template to BLAST short read alignment (allowing 3 mismatches on each side) to identify CDR-H3 seqeunces. Datasets were constructed by retaining only sequences that had at least 5 read counts in at least one panning round or had non-zero reads in all rounds to reduce noise.

E.2 Generating the factorizable library

We used the phage panning data to train reverse kernel models, which we used to generate segment libraries that combine into libraries of size 10^9. SAPS was run for 500 sweeps on a single NVIDIA Titan RTX GPU (24GB RAM) using an ensemble of 20 models to guide the objective, and each sweep took approximately 10 minutes to complete.

Each output segment library contains 5000 sequences of each length between 4 to 10, for a total of 35000 sequences. The resulting factorizable library should have a length distribution similar to the convolution of a pair of uniform distributions, which is shaped like an isosceles triangle. Sequences in the prefix library were padded exclusively on the left while sequences in the suffix library were padded exclusively on the right before being fed to the reverse kernel models, so the model would not detect gaps in the sequences.

As discussed in the section titled "Sequences of different lengths can be represented using padding" in the main text, there may be duplicate sequences, so the libraries may not contain exactly 35000^2 sequences. The number of unique sequences in each library is given in Table S1, along with the λ entropy hyperparameter that was used when generating them.

Library	**Number of unique sequences**	**λ Hyperparameter**
Ranibizumab(+)	1189623032	0.1
Omalizumab(+)	1208884387	0.1
Trastuzumab(+)	1220747116	0.3
Etanercept(+)	1224487103	0.3
BV(-)	1212124707	0.1
Ranibizumab(+) & BV(-)	1193869774	0.1
Ranibizumab(+) & BV(- -)	1214562933	0.1
Ranibizumab(++) & BV(-)	1190050282	0.1

Table S1. Unique sequences in factorizable libraries and their entropy hyperparameter

E.3 λ Hyperparameter tuning

To select λ which is a weight on the entropy term in the objective function for model training, we first generate smaller 10^5 libraries composed of segment libraries that contain 700 segments each (100 segments for each length between 4 to 10 inclusive) at $\lambda = [0, 0.001, 0.01, 0.03, 0.1, 0.3, 1]$. We then evaluate the pairwise Levenshtein distance between members of this library and score the library with the corresponding unrestricted model. Based on this analysis across targets, we recommend setting this hyperparameter between 0.1 and 0.3 based on the intended library diversity or score distribution. We provide an example of this tuning conducted on Ranibizumab(+) in Fig. S1).

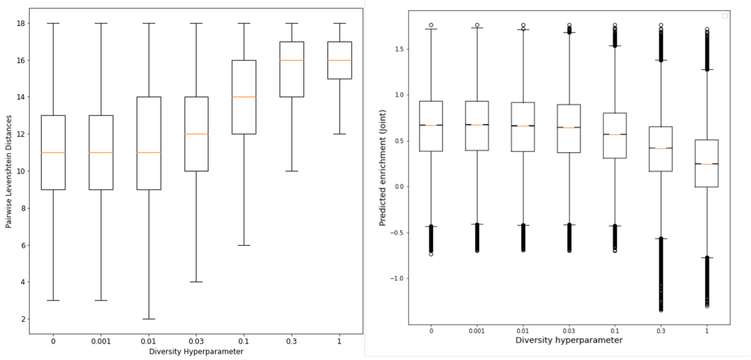

Fig. S1. Ranibizumab(+) λ hyperparameter tuning

F Details on Analyses

F.1 Details of library validation metrics

For all examples of library validation against FW-kappa, we evaluate diversity by measuring the pairwise Levenshtein distance between 10,000,000 uniformly sampled pairs of length 12 sequences. Briefly, the Levenshtein distance between two strings is the minimum number of single-character edits (substitutions, insertions, or deletions) required to change one string into the other. We compute Levenshtein distance using python-Levenshtein v0.12.2. For gigalibrary scoring, 1,000,000 sequences are uniformly sampled and scored with the corresponding unrestricted model to evaluate the sequence optimality of large generated libraries.

F.2 Diversity of prefix and suffix libraries independently

For diversity analysis, we have primarily shown that the diversity of full length SAPS designed CDR-H3 sequences is higher than FW_kappa. To further investigate the source of diversity, we show in Figure S2 that the individual Levenshtein distance distributions of the prefix and suffix libraries are similar across target objectives. Each distribution is calculated from 10,000,000 uniformly sampled segments over all segment lengths. This indicates that the diversity of concatenated libraries is not a result of diversity isolated to the prefix or suffix of the CDR-H3.

F.3 Details on nonspecific motif enrichment analysis

We applied motif enrichment analyses using STREME from the DREME suite [Bailey, 2021]. For all STREME motif enrichment analysis, 490,000 sequences were sampled from both the primary and negative libraries and STREME was run with the following parameters: –protein –minw 3 –maxw 4 –nmotifs 100.

After construction of the BV(-) library optimized for limited polyspecificity, we conducted motif enrichment analysis to check whether known nonspecific motifs were decreased in the BV(-) library when compared to the FW-kappa randomized library. As a preliminary analysis, we computed the number of known tryptophan, valine, arginine, and glycine nonspecific motifs (as identified in [Kelly et al., 2018]) in a sample of

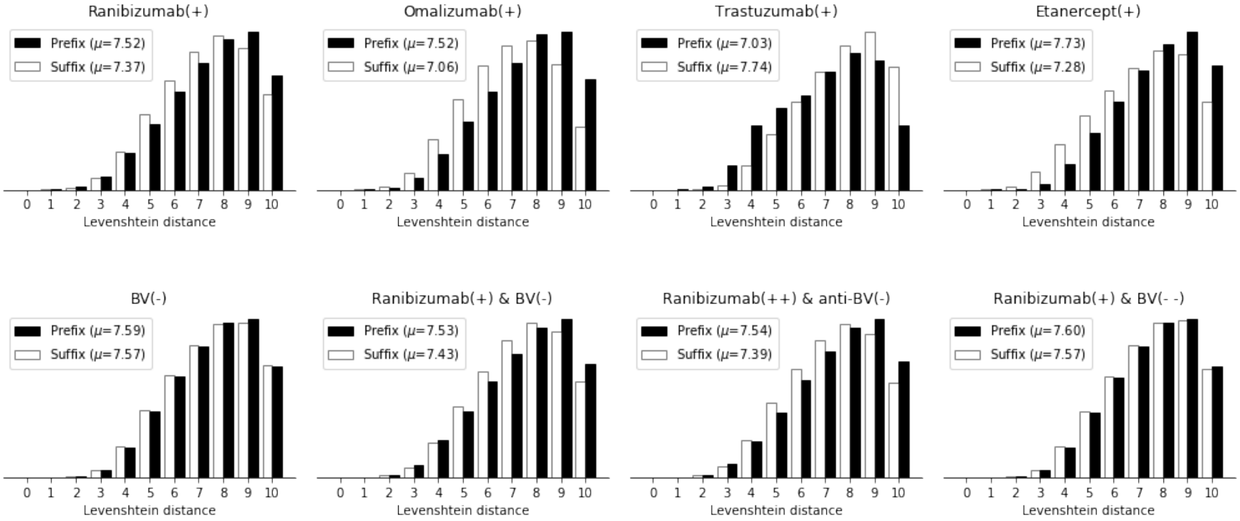

Fig. S2. Prefix and suffix libraries have similar diversity.
For each target evaluated, we compute the pairwise Levenshtein distances of the corresponding prefix (black) and suffix library (white).

490,000 sequences from both libraries (Figure S3). Next, we applied STREME motif enrichment analysis to search for motifs enriched in FW-kappa over the BV library and report a few nonspecific motifs enriched in FW-kappa with significance values (Table S2). We observe that nonspecific motifs are enriched in the FW-kappa library over the BV factorizable library, further suggesting that the library has a favorable developability profile. Finally, we used STREME to identify motifs enriched in the Ranibizumab(++) & BV(-) library over the Ranibizumab(+) & BV(- -) library and identify enriched nonspecific motifs, highlighting the ability to tune factorizable libraries to specific tasks while improving developability (Figure 4D-F).

Motif	**p-value**	**no. of sites**
YYY	3.3e-3456	159437
GRG	1.5e-1007	79625
DVV	6.3e-080	3959
GGHS	1.2e-054	3324
GGD	3.4e-028	1701
WGG	3.8e-014	434
VGVD	1.6e-013	622

Table S2. Nonspecific motifs from STREME output for enriched sequences in FW-kappa over negative BV library

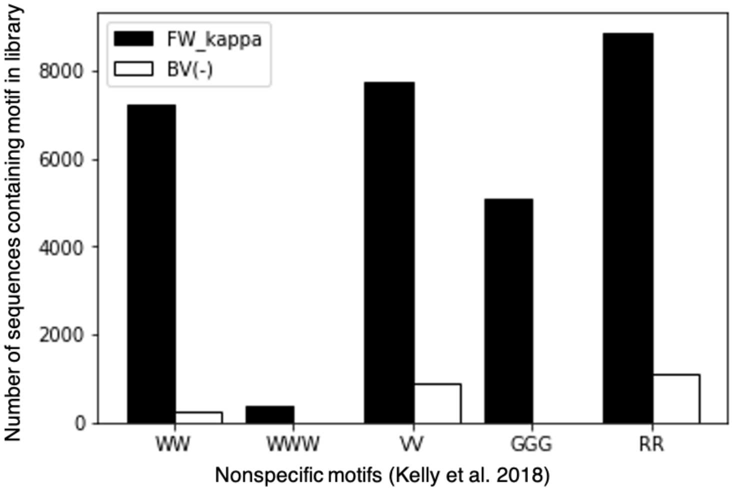

Fig. S3. Nonspecific motif counts in BV(-) vs. FW-kappa

References

[Bailey, 2021] Bailey, T. L., 2021. Streme: accurate and versatile sequence motif discovery. *Bioinformatics*, **37**(18):2834–2840.

[Eddy, 2004] Eddy, S. R., 2004. Where did the blosum62 alignment score matrix come from? *Nature biotechnology*, **22**(8):1035–1036.

[Kelly et al., 2018] Kelly, R. L., Le, D., Zhao, J., and Wittrup, K. D., 2018. Reduction of nonspecificity motifs in synthetic antibody libraries. *J. Mol. Biol.*, **430**(1):119–130.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J., 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, .

[Liu et al., 2020] Liu, G., Zeng, H., Mueller, J., Carter, B., Wang, Z., Schilz, J., Horny, G., Birnbaum, M. E., Ewert, S., and Gifford, D. K., *et al.*, 2020. Antibody complementarity determining region design using high-capacity machine learning. *Bioinformatics*, **36**(7):2126–2133.

[Zuckerman, 2006] Zuckerman, D., 2006. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 681–690.