Supplementary material

Contents

S1 Overview diagram and notation

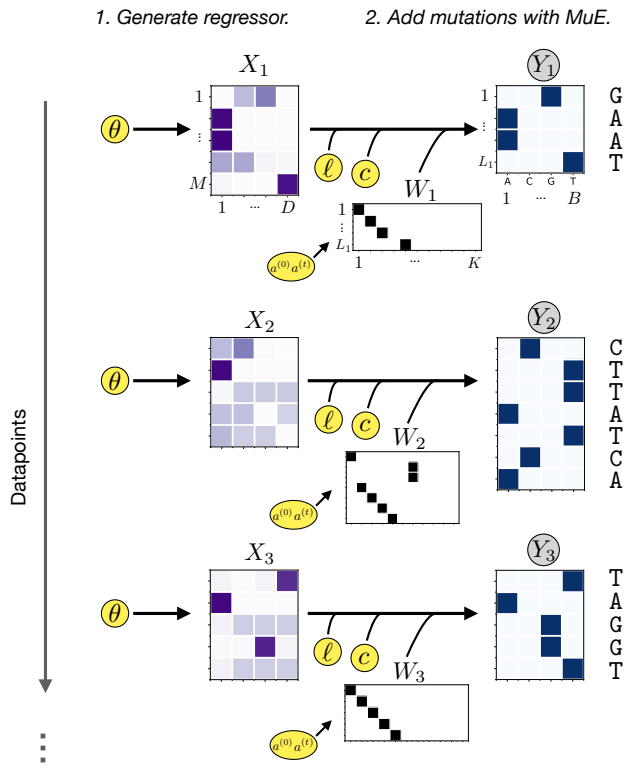

Figure S1: **MuE observation model.** Overview of the generative process in MuE observation models. First, the latent regressor sequence X_i is sampled. Then, the MuE distribution adds mutations to generate Y_i. A latent variable W_i controls the pattern of insertions and deletions. Global parameters that must be inferred are highlighted in yellow.

Table S1: **Notation for MuE observation models** A summary of the notation used in the main text, for convenient reference. Space refers to the space the variable lives in, i.e. $N \in \mathbb{N}$, the set of positive non-zero integers.

Variable	Space	Generation	Description
N	$\mathbb{N}$	Observed	Number of observed sequences.
$\mathcal{B}$	finite set	Hyperparameter	Alphabet (e.g. $\{A, T, G, C\}$ for DNA).
B	$\mathbb{N}$	$B := \lvert\mathcal{B}\rvert$	Alphabet size.
M	$\mathbb{N}$	Hyperparameter	Length of latent regressor sequence. (Typically set to be somewhat larger than $\max_i L_i$.)
D	$\mathbb{N}$	Hyperparameter	Size of latent regressor sequence's alphabet. (Typically set to be somewhat larger than B.)
V_i	$\mathbb{R}^{M \times D}$	$V_i \sim p_\theta$	Output of the initial continuous-space generative model.
X_i	$(\Delta_D)^M$	$X_i := \text{softmax}(V_i)$	Latent regressor sequence, intuitively the "precursor" or "ancestor" to Y_i.
$a^{(0)}$	Δ_K	Parameter	Controls the probability of insertion and deletion mutations occurring in X_i.
$a^{(t)}$	$(\Delta_K)^K$	Parameter	Controls the probability of insertion and deletion mutations occurring in X_i. Must satisfy Condition 2.2.
W_i	$\{1, \ldots, K\}^{L_i}$	$W_i \sim \text{MarkovModel}(a^{(0)}, a^{(t)})$	The hidden Markov model state variable, which defines a latent alignment between X_i and Y_i. (W_i is marginalized out during inference.)
c	$(\Delta_D)^{M+1}$	Parameter	Controls the probability of the insertion sequence letters (but not the presence or absence of the insertion).
ℓ	$(\Delta_B)^D$	Parameter	Substitution matrix.
Y_i	$\mathcal{B}^{L_i}$	$Y_i \sim \text{MuE}(X_i, c, a^{(0)}, a^{(t)})$	Observed sequence, intuitively generated by mutating X_i with substitutions, insertions and deletions.
L_i	$\mathbb{N}$	$L_i := \lvert Y_i\rvert$	Length of observed sequence Y_i.

S2 Theory

S2.1 Illustrating MSA pathologies

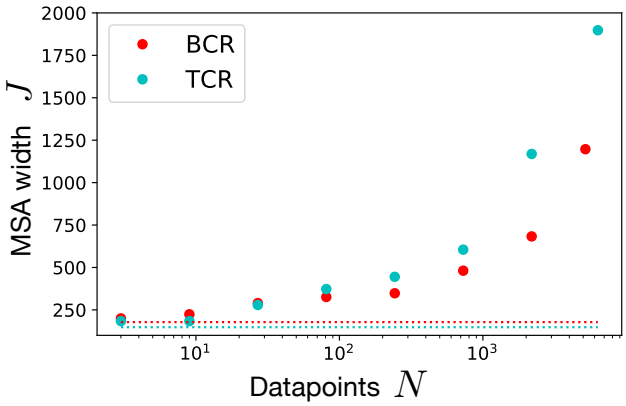

Figure S2: **MSA width can diverge with dataset size.** MSA width J as a function of sequences N in the dataset. BCR is a B cell receptor dataset, TCR a T cell receptor dataset.

To illustrate the problems described in Section 4.1 of the main text, we examined a B cell receptor dataset and a T cell receptor dataset (the 10x Genomics datasets described in Section S6). Sequences were subsampled and aligned using MUSCLE (Edgar, 2004), a standard MSA algorithm. Figure S2 shows the growth in MSA width J as a function of the subsampled dataset size.

S2.2 Proof of Proposition 4.4

To prove the result, we will examine each existing model individually; exact specifications and assumptions for each model are provided in their corresponding section. The probability of the Markov chain terminating given that it is at a state k is denoted $t_k^{(t)}$, and the probability of the Markov chain terminating initially (that is, of the Markov chain taking zero steps) is denoted $t^{(0)}$. Without loss of generality, we will write transition probabilities $a^{(t)}$ and $a^{(0)}$ without conditioning on the Markov chain not terminating, i.e. $\sum_{k'} a_{k,k'}^{(t)} + t_k^{(t)} = 1$. The conditional transition probability can of course be computed as $a_{k,k'}^{(t)}/(1-t_k^{(t)})$. In general, we will also index latent states k of the MuE by their corresponding (m, g) value where (in line with the definition of g_l and m_l) $g = \mathbb{I}(k > M)$ and $m = k - Mg$; we will use k and (m, g) interchangeably for any given state.

It is useful for understanding the following results to have in mind a particular example to illustrate the definitions in the main text.

Sequences	*Pairwise alignment* $\mathcal{A}$	j *and* g *representation*
$Y =$ `ATG`	$\mathcal{A}^{(y)} =$ `A--TG-`	$(j_1, \ldots, j_L) = (1, 4, 5)$
$X =$ `TCTG`	$\mathcal{A}^{(x)} =$ `-TCT-G`	$(g_1, \ldots, g_L) = (1, 0, 1)$

It is also useful to define $m_l := W_l - Mg_l$, which indexes the position within the first or second block of states. For the example we have, $(m_1, \ldots, m_L) = (1, 3, 4)$.

Remark S2.1. *Given sequences X and Y of length M and L respectively, $(j_1, \ldots, j_L)$ and $(g_1, \ldots, g_L)$ uniquely define a pairwise alignment $\mathcal{A}$.*

Proof. Applying Definition 4.2 and the definitions of $(j_1, \ldots, j_L)$ and $(g_1, \ldots, g_L)$ iteratively to each column of the alignment leads to the construction of $\mathcal{A}$ in Algorithm 1. □

Algorithm 1: Pairwise alignment construction

input : $(j_1, \ldots, j_L)$ and $(g_1, \ldots, g_L)$ and X and Y
output: $\mathcal{A}$
$n = 0$ *(indexes position in overall alignment.)*;
$m = 0$ *(indexes position in sequence X)*;
Iterate until each letter in both X and Y has been placed in $\mathcal{A}$;
while $m < M$ ***or*** $n < j_L$ **do**
  $n = n + 1$;
  if $\exists l : n = j_l$ **then**
    $\mathcal{A}_n^{(y)} = Y_l$ *(by definition of j_l)*;
    if $g_l = 1$ **then**
      $\mathcal{A}_n^{(x)} =$ - *(by definition of g_l)*;
    else
      $m = m + 1$;
      $\mathcal{A}_n^{(x)} = X_m$ *(by definitions of g_l and $\mathcal{A}^{(x)}$; letters of X must be in order)*;
    end
  else
    $\mathcal{A}_n^{(y)} =$ - *(by definition of j_l)*;
    $m = m + 1$;
    $\mathcal{A}_n^{(x)} = X_m$ *(by definition of $\mathcal{A}$; each column of $\mathcal{A}$ must have at least one letter)*;
  end
end

S2.2.1 Thorne-Kishino-Felsenstein

The Thorne-Kishino-Felsenstein (TKF) model is a continuous-time stochastic process model of sequence evolution that satisfies detailed balance (Thorne et al., 1991).

Statement Let X be a one-hot encoding of the initial sequence. Let $D = B$ and let π be the TKF parameter corresponding to the equilibrium probability of each letter. For all $m \in \{1, ..., M\}$ and $b \in \{1, ..., B\}$, assign

$$c_{m,b} := \pi_b. \tag{S1}$$

Let $\lambda > 0$ and $\mu > 0$ be the TKF indel rate parameters, with $\lambda < \mu$, and let $\tau > 0$ be the divergence time parameter. Define

$$\beta(\tau) := \frac{1 - e^{-(\mu-\lambda)\tau}}{\mu - \lambda e^{-(\mu-\lambda)\tau}}. \tag{S2}$$

Define the transition matrix and termination probability as

$$a_{k,k'}^{(t)} := \begin{cases} [\mu\beta(\tau)]^{m'-m-1+g} e^{-\mu\tau}[1 - \lambda\beta(\tau)] & \text{if } m - g < m' < M + 1 \text{ and } g' = 0 \\ \lambda\beta(\tau) & \text{if } m - g = m' - 1 \text{ and } g' = 1 \\ [\mu\beta(\tau)]^{m'-m-2+g}[1 - e^{-\mu\tau} - \mu\beta(\tau)][1 - \lambda\beta(\tau)] & \text{if } m - g < m' - 1 \text{ and } g' = 1 \\ 0 & \textit{otherwise.} \end{cases} \tag{S3}$$

$$t_k^{(t)} := [1 - \lambda\beta(\tau)][\mu\beta(\tau)]^{M-m+g} \tag{S4}$$

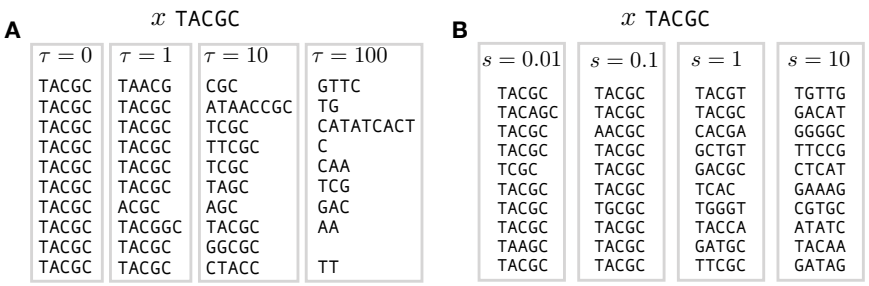

Figure S3: **Samples from the Thorne-Kishino-Felsenstein model.** Initial sequence `TACGC`, $\mu = 0.02$, and $\lambda = 0.01$. A. $s = 0.01$ and varying τ. B. $\tau = 1$ and varying s.

The initial transition vector follows the same form, and can be written as $a_k^{(0)} := a_{0,k}^{(t)}$, and the initial termination probability can be written $t^{(0)} := t_0^{(t)}$ (i.e. they match Equations S3 and S4 with $(m, g) = (0, 0)$ plugged in). Let $s > 0$ be the TKF substitution rate parameter and define the substitution matrix

$$\ell_{b,b'} := \begin{cases} e^{-s\tau} + \pi_{b'}(1 - e^{-s\tau}) & \text{if } b = b' \\ \pi_{b'}(1 - e^{-s\tau}) & \text{if } b \neq b' \end{cases} \tag{S5}$$

With these definitions, $Y \sim \text{MuE}(X, c, \ell, a^{(0)}, a^{(t)})$ is the distribution of the Thorne-Kishino-Felsenstein model after the sequence X evolves for time τ. Note that the limit $\tau \to 0$ is the no-mutation limit. Figure S3 illustrates samples from the TKF model with changing parameters.

Proof We will show that the joint probability of W and Y under the MuE distribution is identical to the joint probability of the corresponding alignment pairwise alignment and Y under the TKF model. To start, we systematically enumerate state transitions in the MuE model and compute the corresponding probability factor under the TKF alignment scoring system. Our alignment notation in this section follows the original paper. "`X`" represents a residue and "`-`" a gap. "`.`" represents the "immortal link" in the model, the start of the sequence. We use "`$`" as a termination symbol. Following the original paper, we define, for $\nu \in \{1, 2, \ldots\}$,

$$\begin{aligned} p_\nu(\tau) &:= e^{-\mu\tau}[1 - \lambda\beta(\tau)][\lambda\beta(\tau)]^{\nu-1} \\ p'_0(\tau) &:= \mu\beta(\tau) \\ p'_\nu(\tau) &:= [1 - e^{-\mu\tau} - \mu\beta(\tau)][1 - \lambda\beta(\tau)][\lambda\beta(\tau)]^{\nu-1} \\ p''_\nu(\tau) &:= [1 - \lambda\beta(\tau)][\lambda\beta(\tau)]^{\nu-1} \end{aligned} \tag{S6}$$

The TKF model assigns probabilities to a pairwise alignment based on the pattern of residues and gaps; we will break down possible pairwise alignments into chunks corresponding to state transitions under the MuE and compute the probability factor that they contribute under the TKF scoring system. When enumerating transitions in the Markov model we put a "|" symbol to the right of the residue we are transitioning from.

1. Transitioning from a state $(m, 0)$ to a state $(m' > m, 0)$ gives the probability factor $[p'_0(\tau)]^{m'-m-1}p_1(\tau) = [\mu\beta(\tau)]^{m'-m-1}e^{-\mu\tau}[1 - \lambda\beta(\tau)]$ according to the TKF scoring system.

   ```
   X | X ... X X
   X | - ... - X
   ```

2. Transitioning from $(m, 1)$ to $(m' \geq m, 0)$ gives the factor $[p'_0(\tau)]^{m'-m}p_1(\tau) = [\mu\beta(\tau)]^{m'-m}e^{-\mu\tau}[1 - \lambda\beta(\tau)]$.

```
- | X ... X X
X | - ... - X
```

3. Transitioning from $(m, 1)$ to $(m, 1)$, situation 1. This gives a factor $\frac{p_{\nu+2}(t)}{p_{\nu+1}(t)} = \lambda\beta(\tau)$.

```
X - ... - | -
X X ... X | X
```

4. Transitioning from $(m, 1)$ to $(m, 1)$, situation 2. This gives a factor $\frac{p'_{\nu+2}(\tau)}{p'_{\nu+1}(\tau)} = \lambda\beta(\tau)$

```
X - ... - | -
- X ... X | X
```

5. Transitioning from $(m, 0)$ to $(m + 1, 1)$. This gives a factor $\frac{p_2(\tau)}{p_1(\tau)} = \lambda\beta(\tau)$.

```
X | -
X | X
```

6. Transitioning from $(m, 0)$ to $(m' > m + 1, 1)$. This gives a factor $[p'_0(\tau)]^{m'-m-2}p'_1(\tau) = [\mu\beta(\tau)]^{m'-m-2}[1 - e^{-\mu\tau} - \mu\beta(\tau)][1 - \lambda\beta(\tau)]$.

```
X | X ... X -
X | - ... - X
```

7. Transitioning from $(m, 1)$ to $(m' > m, 1)$. This gives a factor $[p'_0(\tau)]^{m'-m-1}p'_1(\tau) = [\mu\beta(\tau)]^{m'-m-1}[1 - e^{-\mu\tau} - \mu\beta(\tau)][1 - \lambda\beta(\tau)]$.

```
- | X ... X -
X | - ... - X
```

8. Terminating after $(m, 0)$. This gives a factor $[p'_0(\tau)]^{M-m} = [\mu\beta(\tau)]^{M-m}$.

```
X | X ... X $
X | - ... - $
```

9. Terminating after $(m, 1)$. This gives a factor $[p'_0(\tau)]^{M+1-m} = [\mu\beta(\tau)]^{M+1-m}$.

```
- | X ... X $
X | - ... - $
```

10. Initial transition to $(1, 1)$. This gives a factor $p''_2(\tau) = p''_1(\tau)\lambda\beta(\tau) = [1 - \lambda\beta(\tau)][\lambda\beta(\tau)]$.

```
. | -
. | X
```

11. Initial transition to $(m, 0)$. This gives a factor $p''_1(\tau)[p'_0(\tau)]^{m-1}p_1(\tau) = [1-\lambda\beta(\tau)][\mu\beta(\tau)]^{m-1}e^{-\mu\tau}[1 - \lambda\beta(\tau)]$.

```
. | X ... X X
. | - ... - X
```

12. Initial transition to $(m > 1, 1)$. This gives a factor $p''_1(\tau)[p'_0(\tau)]^{m-2}p'_1(\tau) = [1-\lambda\beta(\tau)][\mu\beta(\tau)]^{m-2}[1-e^{-\mu\tau}-\mu\beta(\tau)][1-\lambda\beta(\tau)]$.

```
. | X ... X -
. | - ... - X
```

13. Terminating in the first step. This gives a factor $[p'_0(\tau)]^M = [\mu\beta(\tau)]^M$.

```
. | X ... X $
. | - ... - $
```

Compiling these results yields the probability factors associated with each transition between states

$$(m,g) \to (m',g') : \begin{cases} [\mu\beta(t)]^{m'-m-1+g}e^{-\mu t}[1-\lambda\beta(t)] \\ \qquad \text{if } m-g < m' < M+1 \text{ and } g' = 0 \\ \lambda\beta(t) \quad \text{if } m-g = m'-1 \text{ and } g' = 1 \\ [\mu\beta(t)]^{m'-m-2+g}[1-e^{-\mu t}-\mu\beta(t)][1-\lambda\beta(t)] \\ \qquad \text{if } m-g < m'-1 \text{ and } g' = 1 \\ 0 \qquad \textit{otherwise} \end{cases} \tag{S7}$$

$$(m,g) \to \textit{termination} : [\mu\beta(t)]^{M-m+g}$$

And with each initial transition

$$\textit{initial} \to (m,g) : \begin{cases} [1-\lambda\beta(t)][\mu\beta(t)]^{m-1}e^{-\mu t}[1-\lambda\beta(t)] & \text{if } 0 < m < M+1 \text{ and } g = 0 \\ [1-\lambda\beta(t)]\lambda\beta(t) & \text{if } m = 1 \text{ and } g = 1 \\ [1-\lambda\beta(t)][\mu\beta(t)]^{m-2}[1-e^{-\mu t}-\mu\beta(t)][1-\lambda\beta(t)] & \text{if } 1 < m \text{ and } g = 1 \\ 0 & \textit{otherwise} \end{cases}$$

$$\textit{initial} \to \textit{termination} : [1-\lambda\beta(t)][\mu\beta(t)]^M \tag{S8}$$

However, these are unnormalized probability factors, not complete probabilities. Note that every alignment will include a factor $[1-\lambda\beta(t)]$, which in the original TKF description is associated with the initial transition. However, if we instead rearrange this factor and assign it to the final transition we obtain the transition matrix given in Equation S3. We can check that this transition matrix normalized. From a state $(m, 0)$, the total outward transition probability is one:

$$\begin{aligned} &\sum_{m'=m+1}^{M} [\mu\beta]^{m'-m-1}e^{-\mu\tau}[1-\lambda\beta] + \lambda\beta + \sum_{m'=m+2}^{M+1} [\mu\beta]^{m'-m-2}[1-e^{-\mu\tau}-\mu\beta][1-\lambda\beta] + [\mu\beta]^{M-m}(1-\lambda\beta) \\ &\quad = \frac{1-(\mu\beta)^{M-m}}{1-\mu\beta}[1-e^{-\mu\tau}-\mu\beta+e^{-\mu\tau}][1-\lambda\beta] + \lambda\beta + [\mu\beta]^{M-m}(1-\lambda\beta) \\ &\quad = 1-(\mu\beta)^{M-m}[1-\lambda\beta] + [\mu\beta]^{M-m}(1-\lambda\beta) \\ &\quad = 1. \end{aligned} \tag{S9}$$

The same expression holds for the initial transition, plugging in $m = 0$. From $(m, 1)$, we have

$$\begin{aligned}
&\sum_{m'=m}^{M} [\mu\beta]^{m'-m} e^{-\mu\tau}[1-\lambda\beta] + \lambda\beta + \sum_{m'=m+1}^{M+1} [\mu\beta]^{m'-m-1}[1-e^{-\mu\tau}-\mu\beta][1-\lambda\beta] + [\mu\beta]^{M+1-m}(1-\lambda\beta) \\
&\quad = \frac{1-(\mu\beta)^{M+1-m}}{1-\mu\beta}[1-e^{-\mu\tau}-\mu\beta+e^{-\mu\tau}][1-\lambda\beta] + \lambda\beta + [\mu\beta]^{M+1-m}(1-\lambda\beta) \\
&\quad = 1-(\mu\beta)^{M+1-m}[1-\lambda\beta] + [\mu\beta]^{M+1-m}(1-\lambda\beta) \\
&\quad = 1.
\end{aligned} \tag{S10}$$

Conditional on the mth residue of X being aligned to the lth residue of Y (i.e. $w_l = m$), the TKF model specifies that the probability of y_l given x_m is $\sum_{b,b'} x_{m,b}\ell_{b,b'}y_{l,b'}$, which is identical to the probability under the MuE model. In the case where the lth residue of y is aligned to a gap (i.e. $g_l = 1$), the TKF model says the probability of choosing the specific base b is π_b, the equilibrium probability of the base. We can check that the MuE provides the same factor:

$$\begin{aligned}
p_{\mathrm{MuE}}(y_{l,b} = 1 | w, x, c, \ell) &= \sum_{b'} c_{m,b'}\ell_{b',b} \\
&= \pi_b e^{-s\tau} + (\pi_b)^2(1-e^{-s\tau}) + \sum_{b''\neq b} \pi_{b''}\pi_b(1-e^{-s\tau}) \\
&= \pi_b e^{-s\tau} + \pi_b(1-e^{-s\tau}) = \pi_b.
\end{aligned} \tag{S11}$$

□

S2.2.2 Pair HMM

The pair HMM model generates pairwise alignments by switching between three states: (1) a state emitting residues in both X and Y (a match state), (2) a state emitting a residue in X and a gap in the alignment of Y, and (3) a state emitting a gap in the alignment of X and a residue in Y (Durbin et al. (1998), Chapter 4.1).

Statement Figure S4 shows a standard pair HMM diagram and state probabilities, with γ the probability of transitioning to a gap state, ϵ the probability of staying in a gap state, and κ the probability of the Markov chain terminating. We assume $1-2\gamma-\kappa \geq 0$ and $1-\epsilon-\kappa \geq 0$. When in a match state, the pair HMM emits letters b and b' in the x and y sequences with probability $\psi_{b,b'}$; otherwise, in gap states, the probability of letter b in the non-gapped sequence is π_b.

Define the MuE transition matrix and termination probability vector as

$$a^{(t)}_{k,k'} := \begin{cases}
\frac{1-2\gamma-\kappa}{1-(\gamma\epsilon^{M-m-1}(1-\kappa)+\kappa+\gamma\kappa\frac{1-\epsilon^{M-m-1}}{1-\epsilon})} & \text{if } m+1 = m' \leq M \text{ and } g = g' = 0 \\
\frac{\gamma\epsilon^{m'-m-2}(1-\epsilon-\kappa)}{1-(\gamma\epsilon^{M-m-1}(1-\kappa)+\kappa+\gamma\kappa\frac{1-\epsilon^{M-m-1}}{1-\epsilon})} & \text{if } m+1 < m' \leq M \text{ and } g = g' = 0 \\
\frac{\gamma}{1-(\gamma\epsilon^{M-m-1}(1-\kappa)+\kappa+\gamma\kappa\frac{1-\epsilon^{M-m-1}}{1-\epsilon})} & \text{if } m+1 = m' \leq M \text{ and } g = 0 \text{ and } g' = 1 \\
\frac{\gamma}{\gamma+\kappa} & \text{if } m+1 = m' = M+1 \text{ and } g = 0 \text{ and } g' = 1 \\
\frac{1-\epsilon-\kappa}{1-\kappa} & \text{if } m = m' \leq M \text{ and } g = 1 \text{ and } g' = 0 \\
\frac{\epsilon}{1-\kappa} & \text{if } m = m' \leq M \text{ and } g = g' = 1 \\
\frac{\epsilon}{\epsilon+\kappa} & \text{if } m = m' = M+1 \text{ and } g = g' = 1 \\
0 & \textit{otherwise}
\end{cases} \tag{S12}$$

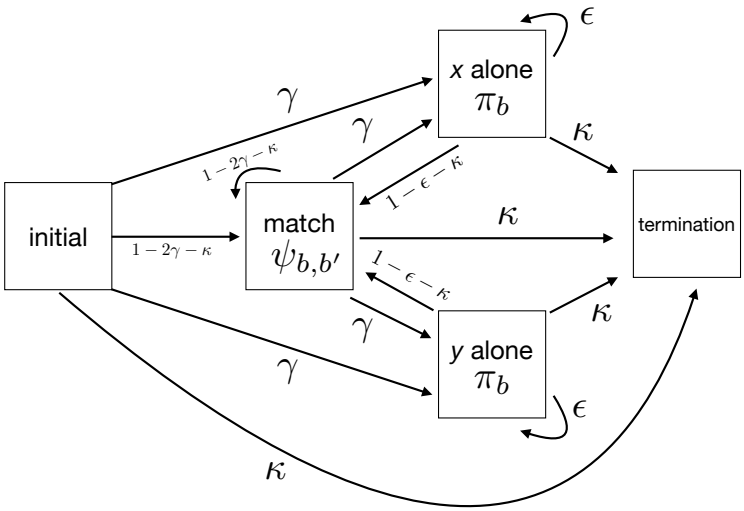

Figure S4: **Pair HMM state diagram.**

$$t_k^{(t)} := \begin{cases} \frac{\gamma\epsilon^{M-m-1}\kappa}{1-(\gamma\epsilon^{M-m-1}(1-\kappa)+\kappa+\gamma\kappa\frac{1-\epsilon^{M-m-1}}{1-\epsilon})} & \text{if } m < M \text{ and } g = 0 \\ \frac{\kappa}{\gamma+\kappa} & \text{if } m = M \text{ and } g = 0 \\ \frac{\kappa}{\epsilon+\kappa} & \text{if } m = M+1 \text{ and } g = 1 \\ 0 & \textit{otherwise} \end{cases} \tag{S13}$$

The initial transition vector is defined by $a_k^{(0)} := a_{0,k}^{(t)}$ and initial termination probability is $t^{(0)} := t_0^{(t)}$. Define the substitution matrix

$$\ell_{b,b'} := \frac{\psi_{b,b'}}{\pi_b} \tag{S14}$$

for all $b, b' \in \{1, ..., B\}$. Let the rows of the insertion matrix c be

$$c_m := (\ell^{-1})^\top \cdot \pi \tag{S15}$$

where ℓ^{-1} is the inverse of the substitution matrix, which is assumed to be an invertible matrix, and $\top$ indicates the matrix transpose.

With these definitions, $Y \sim \text{MuE}(X, c, \ell, a^{(0)}, a^{(t)})$ is equivalent to the conditional distribution of Y given X under the pair HMM. Note that if $\gamma = 0$ and $\psi = \text{diag}(\pi)$ (the $B \times B$ matrix with diagonal entries π and all other entries 0) then we recover the no-mutation limit of the MuE distribution.

Proof We will show that the joint probability of W and Y under the MuE model is identical to the joint probability of the corresponding alignment and Y under the pair HMM, conditional on X. We start by enumerating all possible transitions between states of the MuE Markov chain and computing their probability under the pair HMM model without conditioning on X. Define $\omega_j^x := \mathbb{I}(\mathcal{A}_j^{(x)} \in \mathcal{B})$ and ω^y likewise. We use ω^x, ω^y notation to represent possible alignments, with the symbol "|" placed to the right of the residue we are transitioning *from*.

1. Transitioning from $(m, 0)$ to $(m + 1 \leq M, 0)$ has probability $1 - 2\gamma - \kappa$.

```
x: 1 | 1
y: 1 | 1
```

2. Transitioning from $(m, 0)$ to $(m' > m+1, 0)$ for $m' < M+1$ has probability $\gamma\epsilon^{m'-m-2}(1-\epsilon-\kappa)$.

```
x: 1 | 1 ... 1 1
y: 1 | 0 ... 0 1
```

3. Transitioning from $(m, 0)$ to $(m + 1, 1)$ has probability γ.

```
x: 1 | 0
y: 1 | 1
```

4. Terminating after $(m < M, 0)$ has probability $\gamma\epsilon^{M-m-1}\kappa$.

```
x: 1 | 1 ... 1 $
y: 1 | 0 ... 0 $
```

5. Terminating after $(M, 0)$ has probability κ.

```
x: 1 | $
y: 1 | $
```

6. Transitioning from $(m, 1)$ to $(m \leq M, 0)$ has probability $1 - \epsilon - \kappa$.

```
x: 0 | 1
y: 1 | 1
```

7. Transitioning from $(m, 1)$ to $(m, 1)$ has probability ϵ.

```
x: 0 | 0
y: 1 | 1
```

8. Terminating after $(M + 1, 1)$ has probability κ

```
x: 0 | $
y: 1 | $
```

9. Transitioning from the initial state to $(1, 0)$ has probability $1 - 2\gamma - \kappa$.

```
x: | 1
y: | 1
```

10. Transitioning from the initial state to $(m > 1, 0)$ for $m < M + 1$ has probability $\gamma\epsilon^{m-2}(1 - \epsilon - \kappa)$.

```
x: | 1 ... 1 1
y: | 0 ... 0 1
```

11. Transitioning from the initial state to $(1, 1)$ has probability γ.

```
x: | 0
y: | 1
```

12. Terminating immediately from the initial state has probability $\gamma\epsilon^{M-1}\kappa$ when $M > 0$.

```
x: | 1 ... 1 $
y: | 0 ... 0 $
```

13. Terminating immediately from the initial state has probability κ when $M = 0$.

```
x: | $
y: | $
```

These transition probabilities were derived without conditioning on the fact that we have observed X, which has length M. To compute this conditional probability, we calculate the probability that the pair HMM generates an alignment with too many or too few X residues starting from each MuE Markov model state.

1. Starting from a state $(m < M, 0)$, the probability of the pair HMM generating an invalid alignment that is too long (rather than transitioning to a valid MuE state) is $\gamma\epsilon^{M-m-1}(1-\epsilon-\kappa)+\gamma\epsilon^{M-m} = \gamma\epsilon^{M-m-1}(1-\kappa)$. The first term is from alignments that use a match state instead of terminating.

   ```
   x: 1 | 1 ... 1 1
   y: 1 | 0 ... 0 1
   ```

   The second term is from alignments that use an x-alone state instead of terminating.

   ```
   x: 1 | 1 ... 1 1
   y: 1 | 0 ... 0 0
   ```

2. Starting from a state $(m < M, 0)$, the probability of generating an invalid alignment that is too short (rather than transitioning to a valid MuE state) is $\kappa + \sum_{m'=m+1}^{M-1} \gamma\epsilon^{m'-m-1}\kappa = \kappa + \gamma\kappa\frac{1-\epsilon^{M-m-1}}{1-\epsilon}$. The first term is from alignments that immediately terminate.

   ```
   x: 1 | $
   y: 1 | $
   ```

   The second term is from alignments that terminate early after transitioning to the x-alone state.

   ```
   x: 1 | 1 ... 1 $
   y: 1 | 0 ... 0 $
   ```

3. Starting from the state $(M, 0)$, the probability of generating an invalid alignment is $(1-2\gamma-\kappa)+\gamma = 1-\gamma-\kappa$. The first term is from alignments that use a match state instead of terminating.

   ```
   x: 1 | 1
   y: 1 | 1
   ```

   The second term is from alignments that use an x-alone state instead of terminating.

```
x: 1 | 1
y: 1 | 0
```

4. Starting from a state $(m \leq M, 1)$ the probability of generating an invalid alignment that is too short is κ.

   ```
   x: 0 | $
   y: 1 | $
   ```

5. Starting from the state $(M+1, 1)$, the probability of generating an invalid alignment that is too long is $1 - \epsilon - \kappa$.

   ```
   x: 0 | 1
   y: 1 | 1
   ```

6. Starting from the initial state, the probability of generating an invalid alignment that is too long is $\gamma\epsilon^{M-1}(1-\epsilon-\kappa) + \gamma\epsilon^{M-m} = \gamma\epsilon^{M-1}(1-\kappa)$. The first term is from alignments that use a match state instead of terminating.

   ```
   x: | 1 ... 1 1
   y: | 0 ... 0 1
   ```

   The second term is from alignments that use an x-alone state instead of terminating.

   ```
   x: | 1 ... 1 1
   y: | 0 ... 0 0
   ```

7. Starting from the initial state, the probability of generating an invalid alignment that is too short is $\kappa + \sum_{m'=1}^{M-1} \gamma\epsilon^{m'-1}\kappa = \kappa + \gamma\kappa\frac{1-\epsilon^{M-1}}{1-\epsilon}$ when $M > 0$. The first term is from alignments that immediately terminate.

   ```
   x: | $
   y: | $
   ```

   The second term is from alignments that terminate early after transitioning to the x-alone state.

   ```
   x: | 1 ... 1 $
   y: | 0 ... 0 $
   ```

8. Starting from the initial state, if $M = 0$, then the probability of generating an invalid alignment is $(1 - 2\gamma - \kappa) + \gamma = 1 - \gamma - \kappa$. The first term is from alignments that use a match state.

   ```
   x: | 1
   y: | 1
   ```

   The second term is from alignments that use an x-alone state.

```
x: | 1
y: | 0
```

We can confirm that all possible trajectories of the pair HMM are either valid transitions under the MuE Markov model or produce alignments with too few or too many X residues, by checking that the outward transition probabilities from each state sum to one.

1. From a state $(m < M, 0)$, the total outward transition probability is

$$\begin{aligned}
&(1-2\gamma-\kappa) + \gamma \sum_{m'=m+2}^{M} \epsilon^{m'-m-2}(1-\epsilon-\kappa) + \gamma + \gamma\epsilon^{M-m-1}\kappa + \gamma\epsilon^{M-m-1}(1-\kappa) \\
&+ (\kappa + \gamma\kappa\frac{1-\epsilon^{M-m-1}}{1-\epsilon}) \\
&= 1 - \gamma + \gamma(1-\epsilon-\kappa)\frac{1-\epsilon^{M-m-1}}{1-\epsilon} + \gamma\epsilon^{M-m-1} + \gamma\kappa\frac{1-\epsilon^{M-m-1}}{1-\epsilon} \\
&= 1 - \gamma + \gamma(1-\epsilon^{M-m-1}) + \gamma\epsilon^{M-m-1} \\
&= 1
\end{aligned} \tag{S16}$$

2. From the state $(M, 0)$, the total outward transition probability is

$$\gamma + \kappa + (1-\gamma-\kappa) = 1 \tag{S17}$$

3. From a state $(m \leq M, 1)$, the total outward transition probability is

$$(1-\epsilon-\kappa) + \epsilon + \kappa = 1 \tag{S18}$$

4. From the state $(M+1, 1)$, the total outward transition probability is

$$\kappa + \epsilon + (1-\epsilon-\kappa) = 1 \tag{S19}$$

5. From the initial state, with $M > 0$, the total outward transition probability is

$$\begin{aligned}
&(1-2\gamma-\kappa) + \sum_{m=2}^{M} \gamma\epsilon^{m-2}(1-\epsilon-\kappa) + \gamma + \gamma\epsilon^{M-1}\kappa + \gamma\epsilon^{M-1}(1-\kappa) + (\kappa + \gamma\kappa\frac{1-\epsilon^{M-1}}{1-\epsilon}) \\
&= 1 - \gamma + \gamma(1-\epsilon-\kappa)\frac{1-\epsilon^{M-1}}{1-\epsilon} + \gamma\kappa\epsilon^{M-1} + \gamma\epsilon^{M-1}(1-\kappa) + \gamma\kappa\frac{1-\epsilon^{M-1}}{1-\epsilon} \\
&= 1 - \gamma + \gamma(1-\epsilon^{M-1}) - \gamma\kappa\frac{1-\epsilon^{M-1}}{1-\epsilon} + \gamma\epsilon^{M-1} + \gamma\kappa\frac{1-\epsilon^{M-1}}{1-\epsilon} \\
&= 1
\end{aligned} \tag{S20}$$

6. From the initial state, with $M = 0$, the total outward transition probability is

$$\gamma + \kappa + (1-\gamma-\kappa) = 1 \tag{S21}$$

Consolidating transition probabilities and conditioning on the length of X yields the transition matrix Equation S12.

Next we consider sequence emission probabilities, given an alignment. Recall that X and Y are one-hot encodings of sequences.

1. Consider the case that Y_l is aligned to X_m, ie.

   ```
   x: 1
   y: 1
   ```

   The conditional probability of $Y_{l,b'} = 1$ given $X_{m,b} = 1$ is, according to the pair HMM, $\psi_{b,b'}/\pi_b$. This matches the conditional probability assigned by the MuE,

   $$Y_l \sim \text{Categorical}\big(\sum_{b''} X_{m,b''}\ell_{b''}\big) = \text{Categorical}\big(\frac{\psi_b}{\pi_b}\big). \tag{S22}$$

2. Consider the case that Y_l is aligned to a gap, ie.

   ```
   x: 0
   y: 1
   ```

   The conditional probability of $Y_{l,b}$ given X is just π_b (since X is not informative in this case). This matches the conditional probability assigned by the MuE,

   $$Y_l \sim \text{Categorical}\big((\pi^\top \cdot \ell^{-1} \cdot \ell)^\top\big) = \text{Categorical}(\pi). \tag{S23}$$

3. Consider the case that X_m is aligned to a gap, ie.

   ```
   x: 1
   y: 0
   ```

   The conditional probability of X_m given X is trivially one, so this term does not contribute to the conditional probability of Y given X under the pair HMM. It also does not contribute to the probability under the MuE.

Thus, term-by-term, the joint probability of W and Y under the proposed MuE distribution matches the joint probability of the corresponding alignment and Y under the pair HMM conditional on X. □

S2.2.3 Profile HMM

The profile HMM (pHMM) is a widely used model for defining protein sequence families, inferring multiple sequence alignments, and performing database searches (Durbin et al., 1998).

Statement Define the pHMM insertion parameter $r_{m,j} \in [0,1]$ for all $m \in \{1,...,M+1\}$ and $j \in \{0,1,2\}$, and the deletion parameter $u_{m,j} \in [0,1]$ for all $m \in \{1,...,M\}$ and $j \in \{0,1,2\}$. Then define the MuE transition matrix and termination probability

$$a^{(t)}_{k,k'} := \begin{cases} (1-r_{m+1-g,g})(1-u_{m+1-g,g}) \\ \qquad \text{if } m+1-g = m' \text{ and } g' = 0 \\ (1-r_{m+1-g,g})u_{m+1-g,g}(\prod_{m''=m+2-g}^{m'-1}[(1-r_{m'',2})u_{m'',2}])(1-r_{m',2})(1-u_{m',2}) \\ \qquad \text{if } m+1-g < m' \text{ and } g' = 0 \\ r_{m+1-g,g} \\ \qquad \text{if } m+1-g = m' \text{ and } g' = 1 \\ (1-r_{m+1-g,g})u_{m+1-g,g}(\prod_{m''=m+2-g}^{m'-1}[(1-r_{m'',2})u_{m'',2}])r_{m',2} \\ \qquad \text{if } m+1-g < m' \text{ and } g' = 1 \\ 0 \quad \text{otherwise} \end{cases} \tag{S24}$$

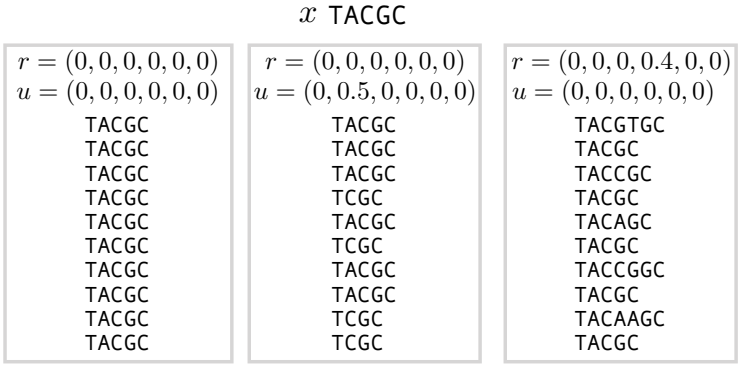

Figure S5: **Samples from the profile HMM.** The regressor sequence $X_{1,...,M}$ is set to TACGC, and we set $r_{m,j=0} = r_{m,j=1} = r_{m,j=2}$ and $u_{m,j=0} = u_{m,j=1} = u_{m,j=2}$ for all m.

$$t_k^{(t)} := \begin{cases} 1 - r_{M+1,g} \\ \qquad \text{if } m - g = M \\ (1 - r_{m+1-g,g}) u_{m+1-g,g} (\prod_{m''=m+2-g}^{M} [(1 - r_{m'',2}) u_{m'',2}]) (1 - r_{M+1,2}) \\ \qquad \text{if } m - g < M \end{cases} \tag{S25}$$

The initial transition vector is given by $a_k^{(0)} := a_{0,k}^{(t)}$ and the initial termination probability is given by $t^{(0)} = t_0^{(t)}$. Let the MuE substitution matrix ℓ be the identity matrix I_B, ie.

$$\ell_{b,b'} := \delta_{b,b'} \tag{S26}$$

for $b, b' \in \{1, ..., B\}$.

With these definitions the profile HMM can be written as $Y \sim \text{MuE}(X, c, \ell, a^{(0)}, a^{(t)})$. Figure S5 illustrates samples from the pHMM. Intuitively, r controls insertion probabilities and u controls deletion probabilities; when $r_{m,j} = 0$ and $u_{m,j} = 0$ for all m and j, we recover the no-mutation limit of the MuE.

Proof This result follows from the relabeling of the profile HMM Markov state architecture with the (m, g) notation (Figure S6). So-called "delete states" in profile HMMs do not generate observations Y_l. To compute the probability of transitioning between two observable states (m, g) and (m', g'), we compute the probability of (1) direct paths between the two states and (2) all possible paths between the two states that go only through deletion states. This yields Equation S24.

The emission probability of each state in the pHMM is set by its associated emission probability vector. Without loss of generality, we can write any emission matrix of the pHMM as $\tilde{x}$ (Definition 2.1) since ℓ is the identity matrix.

□

S2.2.4 Needleman-Wunsch

The Needleman-Wunsch (NW) algorithm is a classic non-probabilistic alignment method (Needleman and Wunsch, 1970).

Summary Let G be the NW gap penalty, which we assume to be negative, and define $u := e^G$.

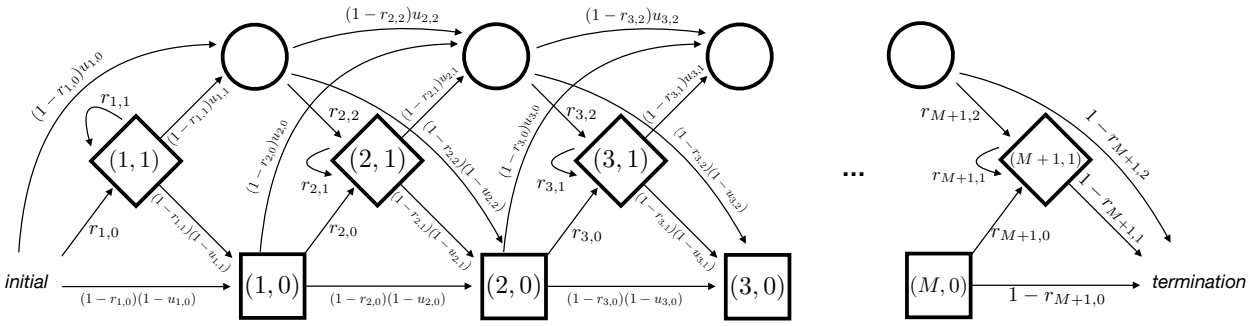

Figure S6: **Profile HMM state architecture.** The conventional profile HMM state architecture labeled with MuE states, using (m, g) notation. Squares indicate "match states", diamonds indicate "insert states", and circles indicate "delete states".

We define the MuE transition matrix and termination probabilities

$$a^{(t)}_{k,k'} := \begin{cases} \frac{1-u}{1+u} u^{m'-m-1+g} & \text{if } m-g < m' < M+1 \text{ and } g' = 0 \\ \frac{1-u}{1+u} u^{m'-m+g} & \text{if } m-g < m' \leq M+1 \text{ and } g' = 1 \\ 0 & otherwise \end{cases} \tag{S27}$$

$$t^{(t)}_k := \frac{1+u^2}{1+u} u^{M-m+g} \tag{S28}$$

The initial transition vector is defined by $a^{(0)}_k := a^{(t)}_{0,k}$ and the initial termination probability is $t^{(0)}_k := t^{(t)}_0$. Let $S_{b,b'}$ be the NW similarity matrix, for which we assume that $\sum_{b'} e^{S_{b,b'}} = B$ for all b. We define, for $b, b' \in \{1, ..., B\}$,

$$\ell_{b,b'} := \frac{e^{S_{b,b'}}}{B}. \tag{S29}$$

Finally, for all $m \in \{1, ..., M+1\}$,

$$c_m := (\ell^{-1})^\top \cdot (1/B, ..., 1/B)^\top \tag{S30}$$

where ℓ^{-1} is the inverse of the substitution matrix (assumed to be invertible) and $(1/B, ..., 1/B)^\top$ is a length B column vector. Let X and Y be the sequences to be aligned.

Under the MuE model $Y \sim \text{MuE}(X, c, \ell, a^{(0)}, a^{(t)})$, the maximum *a posteriori* estimator of the alignment variable w given X and Y corresponds to the Needleman-Wunsch pairwise alignment between X and Y. Note that in the limit $G \to -\infty$ and $S_{b,b'} \to -\infty$ for all $b' \neq b$, we recover the no-mutation limit of the MuE distribution.

Proof We can organize the NW scoring system according to transitions in the MuE Markov model. We use ω^x, ω^y notation to represent alignments, with the symbol "|" placed to the right of the residue we are transitioning *from*. We assign l' to be the residue of Y at the column of the alignment corresponding to state k'.

1. Transitioning from $(m, 0)$ to $(m' > m, 0)$ gives a NW score of $(m'-m-1)G + \sum_{b,b'} x_{m',b} S_{b,b'} y_{l',b'}$.

```
x: 1 | 1 ... 1 1
y: 1 | 0 ... 0 1
```

2. Transitioning from $(m, 0)$ to $(m' > m, 1)$ gives a NW score of $(m' - m)G$

```
x: 1 | 1 ... 1 0
y: 1 | 0 ... 0 1
```

3. Transitioning from $(m, 1)$ to $(m' \geq m, 0)$ gives a NW score of $(m' - m)G + \sum_{b,b'} x_{m',b} S_{b,b'} y_{l',b'}$

```
x: 0 | 1 ... 1 1
y: 1 | 0 ... 0 1
```

4. Transitioning from $(m, 1)$ to $(m' \geq m, 1)$ gives a NW score of $(m' - m + 1)G$.

```
x: 0 | 1 ... 1 0
y: 1 | 0 ... 0 1
```

5. Terminating after $(m, 0)$ gives a NW score of $(M - m)G$.

```
x: 1 | 1 ... 1 $
y: 1 | 0 ... 0 $
```

6. Terminating after $(m, 1)$ gives a NW score of $(M - m + 1)G$.

```
x: 0 | 1 ... 1 $
y: 1 | 0 ... 0 $
```

Now we can rewrite the Needleman-Wunsch objective function in terms of these transitions, rather than in terms of gap and insert scoring. In particular, define

$$\Delta(l', m, g, m', g') := \begin{cases} (m' - m - 1 + g)G + \sum_{b,b'} x_{m',b} S_{b,b'} y_{l',b'} & \text{if } m - g < m' < M \text{ and } g' = 0 \\ (m' - m + g)G & \text{if } m - g < m' \leq M \text{ and } g' = 1 \\ -\infty & otherwise \end{cases} \tag{S31}$$

Based on the cases outlined above, the NW objective function can now be rewritten as

$$\arg\max_{\vec{m},\vec{g}} \sum_{l=1}^{L} \Delta(l, m_{l-1}, g_{l-1}, m_l, g_l) + (M - m_L + g_L)G \tag{S32}$$

where we set $m_0 = 0, g_0 = 0$. If we find the solution to this objective function, then follow the mapping from the list of Markov chain states $(m_1, g_1), ..., (m_L, g_L)$ back to an alignment, we obtain the Needleman-Wunsch alignment between sequences x and y.

Now we examine the maximum *a posteriori* estimator of w under the MuE distribution. We have

$$\arg\max_{w} \log p(y, w|x, c, a, \ell) = \arg\max_{w} \left[\log p(term.|w_L) + \sum_{l=2}^{L} \log p(y_l, w_l|w_{l-1}) + \log p(y_1, w_1) \right] \tag{S33}$$

where $p(term.|w_L)$ is the termination probability after state w_L, which reduces to $p(term.|init.)$ when $L = 0$. Under the given MuE model,

$$p(y_l, w_l|w_{l-1}) = \begin{cases} \frac{1-u}{1+u} u^{m_l - m_{l-1} - 1 + g_{l-1}} \frac{1}{B} \exp(\sum_{b,b'} x_{m_l,b} S_{b,b'} y_{l,b'}) & \text{if } m_{l-1} - g_{l-1} < m_l < M+1 \text{ and } g_l = 0 \\ \frac{1-u}{1+u} u^{m_l - m_{l-1} + g_{l-1}} \frac{1}{B} & \text{if } m_{l-1} - g_{l-1} < m_l \leq M+1 \text{ and } g_l = 1 \\ 0 & otherwise \end{cases} \tag{S34}$$

$$p(term.|w_L) = \frac{1+u^2}{1+u} u^{M - m_L + g_L} \tag{S35}$$

$$p(y_1, w_1) = \begin{cases} \frac{1-u}{1+u} u^{m_1 - 1} \frac{1}{B} \exp(\sum_{b,b'} x_{m_1,b} S_{b,b'} y_{1,b'}) & \text{if } m_1 < M+1 \text{ and } g_1 = 0 \\ \frac{1-u}{1+u} u^{m_1} \frac{1}{B} & \text{if } m_1 \leq M+1 \text{ and } g_1 = 1 \\ 0 & otherwise \end{cases} \tag{S36}$$

$$p(term.|init.) = \frac{1+u^2}{1+u} u^M \tag{S37}$$

Now, the maximum *a posteriori* estimator of w can be written as

$$\begin{aligned} \arg\max_w \log p(y, w|x) &= \arg\max_{\vec{m},\vec{g}} \Big[L \log(\frac{1-u}{1+u}\frac{1}{B}) + \log(\frac{1+u^2}{1+u}) + \sum_{l=1}^{L} \Delta(l, m_{l-1}, g_{l-1}, m_l, g_l) \\ &\qquad + (M - m_L + g_L) G \Big] \\ &= \arg\max_{\vec{m},\vec{g}} \Big[\sum_{l=1}^{L} \Delta(l, m_{l-1}, g_{l-1}, m_l, g_l) + (M - m_L + g_L) G \Big] \end{aligned} \tag{S38}$$

where again $m_0 = 0$ and $g_0 = 0$. This objective function is identical to the NW objective function (Equation S32), so the maximum *a posteriori* estimator of w in the MuE distribution corresponds to the Needleman-Wunsch pairwise alignment of X and Y.

We can confirm that the transition probabilities of the MuE distribution are normalized by considering transitions from state (m, g):

$$\begin{aligned} &\frac{1-u}{1+u} \sum_{m'=m-g+1}^{M} u^{m'-m-1+g} + \frac{1-u}{1+u} \sum_{m'=m-g+1}^{M+1} u^{m'-m+g} + \frac{1+u^2}{1+u} u^{M-m+g} \\ &= \frac{1-u}{1+u} \Big[\sum_{m''=0}^{M-m-1+g} u^{m''} + u \sum_{m''=0}^{M-m+g} u^{m''} \Big] + \frac{1+u^2}{1+u} u^{M-m+g} \\ &= \frac{1}{1+u} [1 - u^{M-m+g} + u - u^{M-m+g+2}] + \frac{1+u^2}{1+u} u^{M-m+g} \\ &= 1 - \frac{1+u^2}{1+u} u^{M-m+g} + \frac{1+u^2}{1+u} u^{M-m+g} \\ &= 1. \end{aligned} \tag{S39}$$

□

Algorithm 2: Multiple sequence alignment construction

input : $\{W_{1,1}, \ldots, W_{1,L_1}\}, \ldots, \{W_{N,1}, \ldots, W_{N,L_N}\}$ and $Y_1, \ldots, Y_N$
output: Y_{MSA}
Plug in definition of j_l and g_l for each sequence;
for $i \in \{1, 2, \ldots, N\}$ **do**
  for $l_i \in \{1, 2, \ldots, L_i\}$ **do**
    $g_{i,l_i} = \mathbb{I}(W_{i,l_i} > M)$;
    $m_{i,l_i} = W_{i,l_i} - M g_{i,l_i}$;
  end
  $g_{i,L_i+1} = 0$ *(for convenience)*;
  $m_{i,L_i+1} = 0$ *(for convenience)*;
end
$n = 0$;
$l_1, l_2, \ldots, l_N = 1$;
Iterate through each latent state, assigning letters of $Y_1, \ldots, Y_N$ to Y_{MSA};
for $\tilde{m} \in \{1, 2, \ldots, M+1\}$ **do**
  Place in the same contiguous set of columns letters generated from the same site in c;
  while $\exists i : m_{i,l_i} = \tilde{m}$ *and* $g_{i,l_i} = 1$ **do**
    $n = n + 1$;
    for $i \in \{1, 2, \ldots, N\}$ **do**
      if $m_{i,l_i} = \tilde{m}$ *and* $g_{i,l_i} = 1$ **then**
        $Y_{\text{MSA},i,n} = Y_{i,l_i}$;
        $l_i = l_i + 1$;
      else
        $Y_{\text{MSA},i,n} =$ -;
      end
    end
  end
  Place in the same column letters generated from the same site in X;
  if $\exists i : m_{i,l_i} = \tilde{m}$ *and* $g_{i,l_i} = 0$ **then**
    $n = n + 1$;
    for $i \in \{1, \ldots, N\}$ **do**
      if $m_{i,l_i} = \tilde{m}$ *and* $g_{i,l_i} = 0$ **then**
        $Y_{\text{MSA},i,n} = Y_{i,l_i}$;
        $l_i = l_i + 1$;
      else
        $Y_{\text{MSA},i,n} =$ -;
      end
    end
  end
end

S2.3 Inferring multiple sequence alignments

In this section we describe how MuE observation models can be used to infer multiple sequence alignments. First we define a multiple sequence alignment, analogously to Definition 4.2.

Definition S2.2 (Multiple sequence alignment)**.** *Let $Y_1, \ldots, Y_N$ be sequences with lengths $L_1, \ldots, L_N$. A multiple sequence alignment $Y_{\text{MSA}} \in (\mathcal{B} \cup \{-\})^J$ has rows $Y_{\text{MSA},1}, \ldots, Y_{\text{MSA},N}$ each consisting of the letters of Y_i, in order, interspersed with gap symbols. The alignment Y_{MSA} must satisfy the condition that for every $j \in \{1, \ldots, J\}$, there exists some $i \in \{1, \ldots, N\}$ such that $Y_{\text{MSA},i,j} \in \mathcal{B}$.*

Consider models of the form of Equation 2, and let W_i be the latent alignment variable associated with sequence Y_i, i.e. $W_{i,1}, \ldots, W_{i,L_i}$ is the path through the latent state space that generated Y_i with length L_i. Algorithm 2 constructs a multiple sequence alignment of the dataset $Y_1, \ldots, Y_N$ given $W_1, \ldots, W_N$, placing $Y_{i,l}$ that are generated from the same state $(m, 0)$ (corresponding to a particular position in the "ancestral" sequence X_i) in the same column. Note in the case of multiple sequence alignments, as opposed to pairwise alignments, there is no longer a unique alignment given W, since X is not observed. The Algorithm 2 construction is chosen to match a standard construction used for the profile HMM (see Durbin et al. (1998), Chapter 6.5), using the fact that the profile HMM is a special case of Equation 2 with $p_\theta(v) = \delta_{v_0}(v)$, where $\delta_{v_0}(v)$ is the Dirac delta function at v_0. In MuE observation models we can apply the same algorithm as for pHMMs, placing $Y_{i,l}$ that are generated from the same state $(m, 0)$ in the same column.

S2.4 Proof of Proposition 4.5

We require that with probability 1, the set $\{j_1, \ldots, j_L\}$ defined by Definition 4.3 is valid, i.e. it must be ordered such that $j_l < j_{l+1}$ for all $l \in \{1, \ldots, L-1\}$. Plugging in Definition 4.3, this is equivalent to the requirement that

$$m_{l+1} > m_l - g_l, \tag{S40}$$

where recall $m_l := W_l - M g_l$. For this inequality to hold with probability 1 for any sample W, Condition 2.2 is necessary and sufficient. □

S2.5 Vogel et al. natural language translation

The Vogel et al. (1996) translation model takes the same general form as a MuE distribution, with X a sentence in one language and Y a sentence in another language (encoded as sequences of words). In particular, with states k indexed by tuples (m, g), the transition matrix takes the form

$$a^{(t)}_{k,k'} := \begin{cases} \frac{r_{M+m'-m}}{\sum_{m''=1}^{M} r_{M+m''-m}} & \text{if } g = g' = 0 \text{ and } m, m' \leq M \\ 0 & \textit{otherwise} \end{cases} \tag{S41}$$

where $r \in \mathbb{R}^{2M}_+$ is a vector of non-negative weights. The initial transition vector is defined by $a^{(0)}_k := a^{(t)}_{0,k}$. The length L of Y is sampled independently of W. We can see that for general r, Condition 2.2 is violated.

S3 Models

In this section we provide a detailed description of the models evaluated in the main text. We parameterized the transition matrix $a^{(t)}$ in terms of r and u following Equation S24 (the profile HMM parameterization). We also considered a simplified variation on Equation S24 where we enforce the constraint $u_{m,0} = u_{m,1} = u_{m,2}$ and likewise $r_{m,0} = r_{m,1} = r_{m,2}$ for all m. We enforced (in both cases) the constraint $u_{M,j} = 0$ for $j \in \{0, 1, 2\}$ (termination has probability zero); rather than assign a termination state we assume the length of the sequence Y_i, that is L_i, is independent of W_i. Since the probability of L_i does not contribute to the per residue perplexity performance

metric (Section S5) we do not use an explicit model for L_i. The initial transition vector followed the same form as the transition matrix, i.e. $a_k^{(0)} = a_{0,k}^{(t)}$.

Note that in our experiments we go slightly beyond the vanilla MuE observation model presented in the main text (Equation 2), and allow the insertion sequence c to also depend on p_θ.

S3.1 Profile HMM

The profile HMM is

$$Y_i \sim \text{MuE}(x, c, \ell = I_B, a^{(0)}(r,u), a^{(t)}(r,u)) \tag{S42}$$

where $a^{(0)}(r,u)$ and $a^{(t)}(r,u)$ depend deterministically on the parameters r and u according to Equation S24, $D = B$, and I_B is the $B \times B$ identity matrix.

S3.2 RegressMuE

The RegressMuE model uses a linear regression model as the MuE observation's continuous-space vector model. Let $H_{i,1}, ..., H_{i,T}$ be covariates associated with sequence Y_i. Let $\beta_0^{(x)}, ..., \beta_T^{(x)} \in \mathbb{R}^{M \times D}$ be a set of coefficients associated with X, and let $\beta_0^{(c)}, ..., \beta_T^{(c)} \in \mathbb{R}^{(M+1) \times D}$ be a set of coefficients associated with c. Then the RegressMuE is

$$\begin{aligned}
V_i^{(x)} &= \beta_0^{(x)} + \sum_{t=1}^{T} H_{i,t}\beta_t^{(x)} \\
V_i^{(c)} &= \beta_0^{(c)} + \sum_{t=1}^{T} H_{i,t}\beta_t^{(c)} \\
Y_i &\sim \text{MuE}(X_i = \text{softmax}(V_i^{(x)}), C_i = \text{softmax}(V_i^{(c)}), \ell, a^{(0)}(r,u), a^{(t)}(r,u)).
\end{aligned} \tag{S43}$$

Note that in this model, unlike the pHMM, the substitution matrix ℓ is not constrained to the identity. When $r_m = q_m = 0$ for all m and $\ell = I_B$, the RegressMuE reduces to a multi-output multinomial logit regression model.

S3.3 FactorMuE

The FactorMuE model is the latent linear version of the RegressMuE. Instead of observing covariates H, we draw a latent variable Z from a standard normal prior,

$$\begin{aligned}
Z_{i,t} &\sim \text{Normal}(0, 1) \\
V_i^{(x)} &= \beta_0^{(x)} + \sum_{t=1}^{T} Z_{i,t}\beta_t^{(x)} \\
V_i^{(c)} &= \beta_0^{(c)} + \sum_{t=1}^{T} Z_{i,t}\beta_t^{(c)} \\
Y_i &\sim \text{MuE}(X_i = \text{softmax}(V_i^{(x)}), C_i = \text{softmax}(V_i^{(c)}), \ell, a^{(0)}(r,u), a^{(t)}(r,u))
\end{aligned} \tag{S44}$$

S3.4 ICAMuE

The ICAMuE model the same as the FactorMuE model, except that it uses a Laplace prior instead of a Normal prior on the local latent variable (Murphy (2012), Chapter 12.6).

$$\begin{aligned}
Z_{i,t} &\sim \text{Laplace}(0,1) \\
V_i^{(x)} &= \beta_0^{(x)} + \sum_{t=1}^{T} Z_{i,t}\beta_t^{(x)} \\
V_i^{(c)} &= \beta_0^{(c)} + \sum_{t=1}^{T} Z_{i,t}\beta_t^{(c)} \\
Y_i &\sim \text{MuE}(X_i = \text{softmax}(V_i^{(x)}), C_i = \text{softmax}(V_i^{(c)}), \ell, a^{(0)}(r,u), a^{(t)}(r,u))
\end{aligned} \tag{S45}$$

S3.5 NeuralMuE

The NeuralMuE model uses a fully connected neural network as the MuE observation's continuous-space vector model. We use a network Γ layers using relu nonlinearities, widths $T_{1:(\Gamma+1)}$, and weights $\beta_{1:(\Gamma+1)}$. Let $H_{i,1:T_{(\Gamma+1)}}$ be a vector of covariates.

$$\begin{aligned}
V_{i,\Gamma+1} &= \beta_{\Gamma+1,0} + \sum_{t=1}^{T_{\Gamma+1}} H_{i,t}\beta_{\Gamma+1,t} \\
V_{i,\Gamma} &= \beta_{\Gamma,0} + \sum_{t=1}^{T_{\Gamma}} \text{relu}(V_{i,\Gamma+1,t})\beta_{\Gamma,t} \\
&\ldots \\
V_{i,1}^{(x)} &= \beta_{1,0}^{(x)} + \sum_{t=1}^{T_1} \text{relu}(V_{i,2,t})\beta_{1,t}^{(x)} \\
V_{i,1}^{(c)} &= \beta_{1,0}^{(c)} + \sum_{t=1}^{T_1} \text{relu}(V_{i,2,t})\beta_{1,t}^{(c)} \\
Y_i &\sim \text{MuE}(X_i = \text{softmax}(V_{i,1}^{(x)}), C_i = \text{softmax}(V_{i,1}^{(c)}), \ell, a^{(0)}(r,u), a^{(t)}(r,u))
\end{aligned} \tag{S46}$$

S3.6 LatentNeuralMuE

The LatentNeuralMuE model uses a neural network latent variable model as the MuE observation's continuous-space vector model. It is the latent covariate version of the NeuralMuE, where instead

of observing H we draw a latent variable Z from a standard normal prior.

$$\begin{aligned}
Z_{i,t} &\sim \text{Normal}(0,1) \\
V_{i,\Gamma+1} &= \beta_{\Gamma+1,0} + \sum_{t=1}^{T_{\Gamma+1}} Z_{i,t}\beta_{\Gamma+1,t} \\
V_{i,\Gamma} &= \beta_{\Gamma,0} + \sum_{t=1}^{T_{\Gamma}} \text{relu}(V_{i,\Gamma+1,t})\beta_{\Gamma,t} \\
&\ldots \\
V_{i,1}^{(x)} &= \beta_{1,0}^{(x)} + \sum_{t=1}^{T_1} \text{relu}(V_{i,2,t})\beta_{1,t}^{(x)} \\
V_{i,1}^{(c)} &= \beta_{1,0}^{(c)} + \sum_{t=1}^{T_1} \text{relu}(V_{i,2,t})\beta_{1,t}^{(c)} \\
Y_i &\sim \text{MuE}(X_i = \text{softmax}(V_{i,1}^{(x)}), C_i = \text{softmax}(V_{i,1}^{(c)}), \ell, a^{(0)}(r,u), a^{(t)}(r,u))
\end{aligned} \tag{S47}$$

S3.7 Priors

We place standard normal priors Normal$(0,1)$ over each element of each coefficient matrix β in each model. Recall that each row of the matrix ℓ is constrained to the simplex, $\ell_d \in \Delta_B$. To enable easy gradient-based optimization and stochastic variational inference (Kucukelbir et al., 2017), we transform an unconstrained parameter $\tilde{\ell} \in \mathbb{R}^{D\times B}$ with a Gaussian prior to the simplex,

$$\begin{aligned}
\tilde{\ell}_{d,b} &\sim \text{Normal}(0,1) \\
\ell_d &= \text{softmax}(\tilde{\ell}_d).
\end{aligned} \tag{S48}$$

The variables $r_{m,j}$ and $u_{m,j}$ are constrained to $[0,1]$ for all m and j. This corresponds to the first dimension of a simplex Δ_2, and so we apply the same approach,

$$\begin{aligned}
\tilde{r}_{m,j,\vartheta} &\sim \text{Normal}(\mu_\vartheta^{(r)}, 1) \text{ for } \vartheta \in \{1,2\} \\
r_{m,j} &= \frac{\exp(\tilde{r}_{m,j,2})}{\exp(\tilde{r}_{m,j,1}) + \exp(\tilde{r}_{m,j,2})}
\end{aligned} \tag{S49}$$

where $\mu^{(r)}$ is a hyperparameter. The variable u_m is handled identically, with prior $\tilde{u}_{m,j,\vartheta} \sim \text{Normal}(\mu_\vartheta^{(u)}, 1)$ for $\vartheta \in \{1,2\}$.

In the case of the ICAMuE model we found that training improved with an annealing strategy: we multiplied each coefficient matrix β by a scalar inverse-temperature parameter ξ, drawn according to $\tilde{\xi} \sim \text{Normal}(100,1)$ and $\xi = \text{softplus}(\tilde{\xi})$ where softplus $= \log(1+\exp(\cdot))$; the variational approximation to ξ (see below) was initialized such that $q(\tilde{\xi})$ had mean 0. Note that this annealing approach does not change the expressivity of the model, only the prior and training dynamics. Details can be found in the supplementary code (see Section S4.2).

S4 Inference

S4.1 Stochastic variational inference

Variational inference approximates the posterior distribution $p(\theta|Y_{1:N})$ of a given probabilistic model using a tractable family of distributions $q_\eta(\theta|Y_{1:N})$ parameterized by η (Blei et al., 2017).

To form this approximation, variational inference minimizes the Kullback-Leibler (KL) divergence between the two distributions,

$$\eta_0 := \arg\min_{\eta} \mathrm{KL}(q_\eta(\theta|Y_{1:N})||p(\theta|Y_{1:N})) \tag{S50}$$

This objective can be rewritten as maximizing the evidence lower bound (ELBO),

$$\eta_0 = \arg\max_{\eta} \mathbb{E}_{q_\eta(\theta|Y_{1:N})}[\log p(Y_{1:N}, \theta)] - \mathbb{E}_{q_\eta(\theta|Y_{1:N})}[\log q_\eta(\theta|Y_{1:N})] = \arg\max_{\eta} \mathrm{ELBO}(\eta) \tag{S51}$$

We employ mean-field variational inference for MuE observation models. We use a diagonal Gaussian distribution, with unknown mean and standard deviation, for the variational distribution over the global parameters $\tilde{r}, \tilde{u}, \tilde{\ell}$, $\tilde{\xi}$ and β. For the local variable z in the FactorMuE and LatentNeuralMuE, we amortize inference using an inference network (also known as an encoder network) (Kingma and Welling, 2014; Rezende et al., 2014). In particular, we set

$$q_{\eta_z}(z_{1:N}|Y_{1:N}) = \prod_{i=1}^{N} q_{\eta_z}(z_i|Y_i) = \prod_{i=1}^{N} \mathcal{N}(z_i|f^{(\mu)}(Y_i;\eta_z), f^{(\sigma)}(Y_i;\eta_z)) \tag{S52}$$

where $\mathcal{N}(z|\mu,\sigma)$ is the probability distribution function of a Gaussian with mean μ and standard deviation σ, and $f^{(\mu)}(Y_i;\eta_z)$ and $f^{(\sigma)}(Y_i;\eta_z)$ are differentiable functions of η_z. We parameterize $f^{(\mu)}$ and $f^{(\sigma)}$ using a neural network,

$$\begin{aligned}
y^{(q)}_{i,l} &= \mathbb{E}_{Y' \sim \mathrm{MuE}(Y_i, c^{(q)}, \ell^{(q)}, a^{(0)}(r^{(q)}, u^{(q)}), a^{(t)}(r^{(q)}, u^{(q)}))}[Y'_l] \\
v^{(q)}_{i,\Gamma^{(q)}+1} &= \beta^{(q)}_{\Gamma^{(q)}+1,0} + \sum_{l=1}^{L^{(q)}} \sum_{b=1}^{B} y^{(q)}_{i,l,b} \beta^{(q)}_{\Gamma^{(q)}+1,l,b} \\
v^{(q)}_{i,\Gamma^{(q)}} &= \beta^{(q)}_{\Gamma^{(q)},0} + \sum_{t=1}^{T_{\Gamma^{(q)}}} \mathrm{relu}(v^{(q)}_{i,\Gamma^{(q)}+1,t}) \beta^{(q)}_{\Gamma^{(q)},t} \\
&\ldots \\
f^{(\mu)} &= \beta^{(q,\mu)}_{1,0} + \sum_{t=1}^{T_1} \mathrm{relu}(v^{(q)}_{i,2,t}) \beta^{(q,\mu)}_{1,t} \\
f^{(\sigma)} &= \left|\beta^{(q,\sigma)}_{1,0} + \sum_{t=1}^{T_1} \mathrm{relu}(v^{(q)}_{i,2,t}) \beta^{(q,\sigma)}_{1,t}\right|.
\end{aligned} \tag{S53}$$

where we have introduced the variational parameters $(\beta^{(q)}, c^{(q)}, r^{(q)}, u^{(q)}, \ell^{(q)}) =: \eta_z$. The first layer of the encoder employs the MuE distribution and computes the expected value of mutants of Y_i, at positions $l \in \{1, ..., L^{(q)}\}$; this expected value is a differentiable function of the MuE parameters, and can be tractably computed using the forward algorithm. We use the same parameterization of the MuE distribution as in the models (Section S3), but fix $r^{(q)}_{1,0} = r^{(q)}_{1,1} = r^{(q)}_{1,2} = r^{(q)}_{2,0} = ... = r^{(q)}_{M,2}$ and $u^{(q)}_{1,0} = u^{(q)}_{1,1} = u^{(q)}_{1,2} = u^{(q)}_{2,0} = ... = u^{(q)}_{M-1,2}$ and $c^{(q)}_1 = c^{(q)}_2 = ... = c^{(q)}_M$. Intuitively, the MuE encoding serves to "smear out" the one-hot encoded sequence Y_i according to learnable insertion, deletion and substitution probabilities, making it easier for the encoder to learn which sequences are similar, and making each encoded sequence $y^{(q)}_i$ the same length $L^{(q)}$.

To optimize the variational approximation we need to compute the gradient of the ELBO with respect to the variational parameters η. To enable faster optimization we employ stochastic variational inference, approximating the gradient at each update step using a minibatch of

data (Ranganath et al., 2014). Let $\phi := (\beta, r, u, \ell)$ be the global parameters of the MuE observation models proposed in Section S3 and let η_ϕ be the parameters of the associated mean-field variational distribution. Then the gradient of the ELBO is

$$\begin{aligned}\nabla_\eta \text{ELBO}(\eta) &= \sum_{i=1}^{N}\Big(\nabla_\eta \mathbb{E}_{q_{\eta_\phi}(\phi)q_{\eta_z}(z_i|Y_i)}[\log p(Y_i|Z_i,\phi)] + \nabla_\eta \mathbb{E}_{q_{\eta_z}(z_i|Y_i)}\Big[\log \frac{p(Z_i)}{q_{\eta_z}(Z_i|Y_i)}\Big]\Big) \\ &\quad + \nabla_\eta \mathbb{E}_{q_{\eta_\phi}(\phi)}\Big[\log \frac{p(\phi)}{q_{\eta_\phi}(\phi)}\Big] \\ &\approx \frac{N}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}\Big(\nabla_\eta \mathbb{E}_{q_{\eta_\phi}(\phi)q_{\eta_z}(z_i|Y_i)}[\log p(Y_i|Z_i,\phi)] + \nabla_\eta \mathbb{E}_{q_{\eta_z}(z_i|Y_i)}\Big[\log \frac{p(Z_i)}{q_{\eta_z}(Z_i|Y_i)}\Big]\Big) \\ &\quad + \nabla_\eta \mathbb{E}_{q_{\eta_\phi}(\phi)}\Big[\log \frac{p(\phi)}{q_{\eta_\phi}(\phi)}\Big]\end{aligned} \tag{S54}$$

where $\mathcal{S} \subseteq \{1, ..., N\}$ is the set of datapoint indices making up the minibatch and $|\mathcal{S}|$ is the size of the set $\mathcal{S}$. We estimate the gradient of the first term on the right hand side of this equation using the reparameterization trick Monte Carlo estimator (with a single sample) and automatic differentiation (Kucukelbir et al., 2017; Kingma and Welling, 2014; Rezende et al., 2014). The remaining terms can be computed analytically (see e.g. Kingma and Welling (2014); Rezende et al. (2014)). Note that this approach relies crucially on the fact that the marginal likelihood of the MuE model, $p_{\text{MuE}}(y|x, c, \ell, a^{(0)}, a^{(t)}) = \sum_w p_{\text{MuE}}(y|w, x, c, \ell, a^{(0)}, a^{(t)})$, is a differentiable function of x, c, a and ℓ. We integrate over all possible values of the Markov chain state variable w using the forward algorithm.

It is useful in some circumstances to reweight the variational objective to reduce the amount of regularization placed on the local latent variable. In particular, for $\chi \in [0, 1]$, we reweight the ELBO as

$$\begin{aligned}\text{ELBO}_\chi(\eta) &= \sum_{i=1}^{N}\Big(\mathbb{E}_{q_{\eta_\phi}(\phi)q_{\eta_z}(z_i|Y_i)}[\log p(Y_i|Z_i,\phi)] + \chi\mathbb{E}_{q_{\eta_z}(z_i|Y_i)}\Big[\log \frac{p(Z_i)}{q_{\eta_z}(Z_i|Y_i)}\Big]\Big) \\ &\quad + \mathbb{E}_{q_{\eta_\phi}(\phi)}\Big[\log \frac{p(\phi)}{q_{\eta_\phi}(\phi)}\Big].\end{aligned} \tag{S55}$$

We achieved improved training performance by annealing the weight χ from 0 to 1 linearly over the course of an initial time period during training (Bowman et al., 2016). To avoid posterior collapse and produce informative latent representations, we found it useful in certain cases to anneal χ only up to a low value $\chi_0 << 1$ in which case we are approximating the maximum likelihood estimator of z; this annealing schedule was only used for producing data visualizations, rather than prediction of held out data (Section S8) (Alemi et al., 2018).

S4.2 Probabilistic programming

We implemented a MuE distribution in both Pyro (Bingham et al., 2019) and Edward2 (Tran et al., 2018), probabilistic programming languages that are GPU-enabled and can use a variety of different inference procedures including both stochastic variational inference and MCMC methods. Probabilistic programming systems make it easy to try out different priors and different continuous-space matrix models p_θ; they also make it easy to build joint models of sequences and other types of data.

Documentation for the Pyro implementation can be found at https://docs.pyro.ai/en/dev/contrib.mue.html. Example Pyro models can be found at https://github.com/pyro-ppl/pyro/

tree/dev/examples/contrib/mue. The Edward2 implementation, along with a brief tutorial, is available at https://github.com/debbiemarkslab/MuE.

S5 Evaluation

The per residue perplexity of a probabilistic sequence model $p(y)$, over a dataset $Y_{1:N}$, is defined as

$$\Omega := \exp\big(-\frac{1}{N}\sum_{i=1}^{N}\frac{1}{L_i}\log p(Y_i|L_i)\big). \tag{S56}$$

In evaluating our models, we computed the average log likelihood performance on a heldout test set $Y_{\mathcal{T}}$ for the model distribution learned from the training set $Y_{\mathcal{D}}$. More precisely, we use

$$\hat{\Omega} := \exp\big(-\frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}}\frac{1}{L_i}\mathbb{E}_{q(\phi|Y_{\mathcal{D}})}[\log p(Y_i|L_i,\phi)]\big) \tag{S57}$$

where $q(\phi|y_{\mathcal{D}})$ is the variational approximation to the posterior distribution from the training dataset and $|\mathcal{T}|$ is the size of the test set. For models with local latent variables z_i, we approximate the marginal likelihood using the ELBO (Blei et al., 2017),

$$\hat{\Omega} \approx \exp\left(-\frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}}\frac{1}{L_i}\big(\mathbb{E}_{q(\phi|Y_{\mathcal{D}})q(z_i|Y_i)}[\log p(Y_i|L_i,Z_i,\phi)] + \mathbb{E}_{q(z_i|Y_i)}\Big[\log\frac{p(Z_i)}{q(Z_i|Y_i)}\Big]\big)\right). \tag{S58}$$

We use Monte Carlo estimation for the expectations. In comparing between different models p_1 and p_2, we also report the log Bayes factor associated with the held out data, ie. the difference in total log probability of the heldout data between the two models,

$$\log \mathrm{BF}_{1,2} := \sum_{i\in\mathcal{T}}\mathbb{E}_{q_2(\phi|Y_{\mathcal{D}})}[\log p_2(Y_i|L_i,\phi)] - \sum_{i\in\mathcal{T}}\mathbb{E}_{q_1(\phi|Y_{\mathcal{D}})}[\log p_1(Y_i|L_i,\phi)] \tag{S59}$$

where q_1 and q_2 are the variational approximations associated with p_1 and p_2. For models with local latent variables, we can use the ELBO approximation as in Equation S58. The Bayes factor provides a measurement of the total evidence in favor of one model versus another.

Per residue perplexity is a useful performance metric for biological sequence models because it is an absolute scale and comparable across datasets as well as models. Since per residue perplexity is not yet widely used in the biological literature, in the interest of making it more interpretable we computed the expected per-residue perplexity for a variety of different protein sequence models, covering different data regimes. In particular, for each model $p(y)$, we examined the expected perplexity in the large data limit, assuming that the model is true,

$$\Omega_0 := \exp\big(-\mathbb{E}_{p(y)}\big[\frac{1}{L}\log p(Y|L)\big]\big). \tag{S60}$$

The expected perplexity is the exponentiated entropy of the model distribution, and so also provides a measurement of sequence diversity under the model. Below, we compute the expected perplexity for distributions ranging from the very high diversity regime (all of evolution) down to the very small diversity regime (human population genetics).

Naive

A naive model assigns an equal probability to each amino acid. In this case the per residue perplexity is

$$\Omega_0 = \exp(-\mathbb{E}[\log(1/20)]) = 20. \tag{S61}$$

Amino acid frequencies

A simple modeling approach is to predict individual amino acids solely based on their naturally occurring frequency across evolution. Using the UniprotKB amino acid frequencies f_b for $b \in \{1, ..., B = 20\}$, we have

$$\Omega_0 = \exp\big(-\mathbb{E}_{Y\sim\text{Categorical}(f)}[\log(f^\top \cdot Y)]\big) = \exp(-\sum_{b=1}^{20} f_b \log f_b) \approx 17.92 \tag{S62}$$

where Y is a one-hot encoding (UniProt Consortium, 2019; Gasteiger et al., 2005).

BLOSUM62

If we are studying specific evolutionary families of proteins, an idealized strategy for building a model is to infer the sequence of the last common ancestor and then predict family members using the standard BLOSUM62 substitution matrix (Henikoff and Henikoff, 1992). The BLOSUM62 matrix is a renormalized copula density, but we can convert it into a mutation probability matrix ℓ by assuming the marginal probability of each amino acid follows the UniprotKB frequency across evolution:

$$\begin{aligned}\log \ell_{b,b'} &= \log p(y_{b'} = 1 | x_b = 1) = \log\left(\frac{f_{b,b'}}{f_b}\right) = \log f_{b'} + \log\left(\frac{f_{b,b'}}{f_b f_{b'}}\right) \\ &= \log f_{b'} + \frac{\log(2)}{2}\text{BLOSUM62}_{b,b'}\end{aligned} \tag{S63}$$

where x is a one-hot encoding of the ancestral amino acid, y is a one-hot encoding the mutated amino acid, and $f_{b,b'}$ is the joint probability of amino acids b and b', where $b, b' \in \{1, ..., B = 20\}$. (The $\log(2)/2$ factor comes from the definition of BLOSUM62.) We renormalize the rows ℓ_b to ensure $\ell_b \in \Delta_B$ (BLOSUM62 uses only small integers, producing non-negligible rounding error). Next, we assume that the ancestral sequence is known exactly, has infinite length, and the frequency of each amino acid within the ancestral sequence matches the UniprotKB overall frequency across evolution. The expected per residue perplexity is then

$$\Omega_0 = \exp(-\mathbb{E}_{X\sim\text{Categorical}(f)}\left[\mathbb{E}_{Y\sim\text{Categorical}(X\cdot\ell)}[\log(X^\top \cdot \ell \cdot Y)]\right]) \approx 11.00. \tag{S64}$$

Human Population Genetics

Finally, we examined a simple model of human population variation. Each human has on average roughly 5 million single nucleotide polymorphisms (SNPs) relative to the reference genome (1000 Genomes Project Consortium et al., 2015). Naively assuming a constant mutation rate over the genome, the probability of a mutation occurring in any particular codon is $q_{\text{codon}} = 1-(1-5/6400)^3$, since there are 6.4 billion total base pairs. If we very naively assume a uniform probability of the codon mutating to any other amino acid, then we can use the substitution matrix ℓ defined by

$$\ell_{b,b'} = \begin{cases} \frac{q_{\text{codon}}}{19} & \text{if } b \neq b' \\ 1 - q_{\text{codon}} & \text{if } b = b'. \end{cases} \tag{S65}$$

If we further very naively assume that there are no correlations among mutations at different genome locations when looking across individuals, then the expected per residue perplexity of the sequence distribution is

$$\Omega_0 = \exp\left(\mathbb{E}_{Y\sim\text{Categorical}(x^\top\cdot\ell)}[\log(x^\top \cdot \ell \cdot Y)]\right) \approx 1.024. \tag{S66}$$

S6 Predictive Performance

S6.1 Survey

Dihydrofolate reductase (DHFR) is a widely conserved enzyme, serine recombinase (PINE) is used as a tool for genomic engineering, cyclin dependent kinase inhibitor 1B (CDKN1B/p27) is a cell cycle inhibitor, and the human papillomavirus E6 protein (VE6) is an oncogenic viral protein Hopf et al. (2017); Toth-Petroczy et al. (2016); Tamarozzi and Giuliatti (2018). Evolutionarily related sequences for each were collected using jackhmmer (v3.1) from the UniRef100 dataset (date 6/2019) (Johnson et al., 2010; Eddy, 2011; Suzek et al., 2015). We used seed sequences with Uniprot identifiers DYR_HUMAN (DHFR dataset), PINE_ECOLI (PINE dataset), CDN1B_HUMAN (CDKN1B dataset), and VE6_HPV16 (VE6 dataset). Note that CDKN1B and VE6 have regions classified as disordered. We set a bitscore threshold of 0.5 bits/residue as in Hopf et al. (2017) and ran the jackhmmer search using the API from the EVcouplings package (Hopf et al., 2019). We included the full envelope of the profile HMM hit in the final dataset. The CDN1B dataset had 1,055 sequences and the VE6 dataset 1,609 sequences. We found 32,510 and 79,354 hits respectively for the DHFR and PINE datasets, which we randomly subsampled to 10,000 sequences to create the final datasets. Note that the jackhmmer search algorithm uses a profile HMM to find distant homologs, and thus may bias the dataset to look more like samples from a pHMM; we therefore expect the performance gains from using other MuE observation models, as compared to the pHMM, on these datasets to be smaller (more conservative) than the performance gains that might be achieved on alternative datasets assembled using different search methods. The TCR dataset was not assembled using jackhmmer. Instead, we downloaded a public dataset from 10x Genomics of 6,327 TCR sequences found in CD8+ cytotoxic T-cells https://support.10xgenomics.com/single-cell-vdj/datasets/2.2.0/vdj_v1_hs_cd8_t (download file dated July 28, 2018). These were sequenced using single cell sequencing of peripheral blood mononuclear cells obtained from an individual healthy donor. Internal stop codons were removed from the sequence.

We set the latent alphabet size $D = 25$. In each experiment, we set M to be 10% longer than the longest sequence in the dataset. We used $T = 5$ latent space dimensions in the FactorMuE and layer sizes $T_2 = 5$, $T_1 = 10$ in the LatentNeuralMuE (we found a substantial dropoff in performance when increasing network width or depth). In the recognition network, we set $L^{(q)} = M - 1$. We also used $\Gamma^{(q)} = 0$ (no relu nonlinearities) in the FactorMuE recognition network and $\Gamma^{(q)} = 1$, $T_1 = 10$ in the LatentNeuralMuE recognition network. For the MuE, we used the constraint $u_{m,0} = u_{m,1} = u_{m,2}$ and likewise $r_{m,0} = r_{m,1} = r_{m,2}$ for all m. For the prior on the MuE insertion and deletion parameters we used $\mu^{(r)} = \mu^{(u)} = (100, 1)$ to disfavor indels.

In these particular experiments, models were implemented in PyTorch, with variational inference implemented by hand and without the parallelized forward algorithm (experiments in Sections S6.2 and S6.3 were performed second with the Pyro implementation). We optimized the variational approximation using Adam (Kingma and Ba, 2015) and a minibatch size of 5. The mean of the variational distribution was initialized at the prior mean, while the variance was initialized to a small random value (the absolute value of a sample from a normal distribution with standard deviation 0.01). We used one Monte Carlo sample to estimate the ELBO gradient at each step. For each model and dataset, we evaluated two different learning rates, 0.1 and 0.01, and three different random restarts, selecting among training runs the parameter values that reached the highest ELBO on the training set for making predictions. For models with local latent variables (the FactorMuE and LatentNeuralMuE), we annealed the ELBO reweighting factor χ from 0 to 1 linearly over the first 2 epochs. We trained for 4 epochs total on the DHFR and PINE datasets, and 7 epochs total

on the smaller CDKN1B, VE6 and TCR datasets, which was sufficient for convergence in each model. We estimated the heldout perplexity using one independent Monte Carlo sample per batch. Computations were performed on graphics processing units (NVIDIA Tesla M40, K80 and V100 GPUs), with double precision, and we used gradient accumulation to reduce memory usage. Single training runs ranged from ∼30 min. for smaller datasets (CDKN1B and VE6) to ∼2.5 hours for larger datasets (DHFR, PINE and TCR).

S6.2 Patient immune repertoires

We considered six datasets. "HC 1" consisted of 5,179 BCR sequences from a healthy donor, obtained with single cell sequencing of peripheral blood mononuclear cells, available from 10x Genomics https://support.10xgenomics.com/single-cell-vdj/datasets/3.0.0/vdj_v1_hs_pbmc2_b (download file dated November 15, 2018). The rest of the datasets all were taken from a study of T cell receptors in patients with and without multiple sclerosis during pregnancy (Ramien et al., 2019). Sequences were translated to amino acids based on the provided nucleotide sequence annotations. The dataset "HC 2" is from a healthy patient, third trimester, CD8+ cells. "HC 3" is from a healthy patient, third trimester, CD4+ cells. "MS 1" is from a patient with MS, before pregnancy, CD8+ cells. "MS 2" is from a patient with MS, second trimester, CD8+ cells. "MS 3" is from a patient with MS, third trimester, CD4+ cells. Each of the datasets from Ramien et al. (2019) was uniformly subsampled to 20,000 sequences. Across all datasets, internal stop codons were modeled along with the 20 amino acids (i.e. $B = 21$).

We again set the latent alphabet size to $D = 25$. We set $M = 200$, longer than most sequences in each dataset. We used $T = 5$ latent dimensions in the ICAMuE. In the recognition network we used $\Gamma^{(q)} = 0$ and set $r^{(q)}, u^{(q)}$ and $\ell^{(q)}$ to the no-mutation limit (avoiding the need for the forward algorithm, to speed up inference at some cost in flexibility). We did *not* use either the constraint $u_{m,0} = u_{m,1} = u_{m,2}$ or the constraint $r_{m,0} = r_{m,1} = r_{m,2}$ in these experiments. For the prior on the MuE insertion and deletion parameters we used $\mu^{(r)} = \mu^{(u)} = (10, 0)$, for both the ICAMuE and pHMM models. We used the $\tilde{\xi} \sim \text{Normal}(100, 1)$ prior as described in Section S3.7 for the ICAMuE model.

Models were implemented in Pyro. We used Pyro's stochastic variational inference method (in particular, `JitTrace_ELBO`, the jit-compiled ELBO), and the parallelized forward algorithm (Särkkä and García-Fernández, 2020). Optimization was performed with Adam, with a learning rate of 0.01, and a minibatch size of 5. Initialization was performed the same as previously, with the exception that $q(\tilde{\xi})$ was initialized to have mean zero. Pyro's low-variance ELBO gradient estimators enabled more reliable inference, and so we only used one initialization in each experiment (rather than three). For the HC 1 dataset we trained for 10 epochs, annealing χ for the first 4; for the remaining (larger) datasets, we trained for two epochs, annealing for 1. This was sufficient for convergence. We used the same GPU hardware as previously, but did not use gradient accumulation. Training took ∼20 min. on the larger datasets (the Pyro implementation offers considerable speedup advantages, thanks in part to the parallelized filtering algorithm).

S6.3 Disordered proteins

Toth-Petroczy et al. (2016) collected datasets of evolutionarily related sequences using jackhmmer on the Uniref and Uniprot databases, starting from regions of human proteins classified as disordered. They developed a (heuristic) alignment uncertainty score to determine whether the MSA provided by jackhmmer was trustworthy enough to apply a Potts model and reach conclusions about epistatic interactions between positions in the MSA. They did not proceed with the Potts

model analysis on datasets with a sufficiently high uncertainty score; we examined these datasets in particular (https://marks.hms.harvard.edu/disorder/proteome). We focused on moderately sized datasets: those with more than 3,000 but less than 25,000 sequences, with the disordered segment less than 160 amino acids long. As before, we included the full envelope of the profile HMM hit in the final dataset.

We used the same hyperparameters and training procedure as in Section S6.2, but set the number of epochs to be the minimum number such that at least 50,000 optimization steps were taken, and the number of epochs of χ annealing to half this number (rounded up).

Detailed results Perplexity on a randomly held out 20% of sequences are shown in Table S2. In 55 out of the 56 datasets, the relative performance of the pHMM and ICAMuE on the training data accurately reflected their relative performance on the test set, i.e. when the pHMM outperformed the ICAMuE model on the training set it also did so on the test set and vice versa. The ICAMuE seems to offer particular advantages when the pHMM itself has low perplexity: among datasets with pHMM perplexity below 8, we find the ICAMuE performs better in more than half (16 out of 31), while among datasets with pHMM perplexity below 5, the ICAMuE performs better in 5 out of 6.

Table S2: Heldout perplexity on disordered protein datasets. "Disordered segment" is the region of the protein classified as disordered that was used as a seed in jackhmmer. "Size" is the total number of sequences in the dataset. Rows sorted by pHMM perplexity.

Gene name	Uniprot id	Disordered segment	Size (sequences)	pHMM	ICAMuE
AKAP6	Q13023	293-431	6349	2.88	**1.98**
NSD1	Q96L73	2463-2590	6517	2.93	**2.64**
NFAT5	O94916	633-769	10283	2.94	**1.98**
CIC	Q96RK0	48-207	7511	**3.10**	3.79
S26A8	Q96RN1	847-970	9466	4.14	**2.67**
TADBP	Q13148	261-373	12873	4.78	**2.97**
TET2	Q6N021	1475-1587	22017	**5.11**	5.98
K2022	Q5QGS0	589-707	3719	**5.14**	5.99
YAF2	Q8IY57	53-180	16005	**5.42**	5.98
HDAC5	Q9UQL6	479-631	14275	**5.44**	5.85
MUC19	Q7Z5P9	5890-6021	13491	5.59	**4.84**
RBM27	Q9P2N5	91-247	11685	**5.80**	6.28
DEN1A	Q8TEH3	453-567	6070	**5.84**	6.11
K1683	Q9H0B3	383-502	10098	5.94	**4.39**
FNBP1	Q96RU3	280-432	23781	5.96	**3.82**
TOX	O94900	135-269	9881	**6.02**	6.48
SRPK3	Q9UPE1	238-348	9345	6.06	**5.73**
CAC1G	O43497	470-626	16502	**6.16**	7.07
NGAP	Q9UJF2	803-953	6356	6.39	**4.45**
PS1C1	Q9UIG5	1-126	3434	6.62	**6.13**
GPKOW	Q92917	31-157	5888	6.67	**4.90**
GOG8B	A8MQT2	1-131	3674	**7.17**	8.04
CPXM1	Q96SM3	30-137	3538	**7.28**	11.51

Continued on next page

Table S2 – continued from previous page

Gene name	Uniprot id	Disordered segment	Size (sequences)	pHMM	ICAMuE
ESX1	Q8N693	34-147	10234	**7.34**	10.44
PPIL4	Q8WUA2	337-492	5897	7.38	**6.06**
TAOK3	Q9H2K8	316-433	8661	**7.38**	8.32
CAAP1	Q9H8G2	197-335	19715	7.54	**5.00**
CCD66	A2RUB6	681-830	9586	**7.66**	9.02
GCC2	Q8IWJ2	1416-1552	7593	7.74	**5.71**
ASXL3	Q9C0F0	107-236	5108	**7.75**	8.42
ARHGF	O94989	273-413	4290	7.97	**7.66**
YJ013	Q6ZQT7	1-158	22994	8.07	**4.32**
PHLB2	Q86SQ0	842-976	22091	**8.34**	10.61
CC168	Q8NDH2	86-232	12240	**8.40**	9.73
41	P11171	690-805	7429	**8.70**	10.05
CEBPA	P49715	161-314	20149	**8.81**	10.18
CP250	Q9BV73	2213-2346	17867	**9.08**	12.34
CHD6	Q8TD26	2312-2457	8843	**9.19**	10.62
ANKH1	Q8IWZ3	2000-2149	15540	**9.22**	10.59
CPLX4	Q7Z7G2	18-128	20000	**9.42**	11.16
WAC	Q9BTA9	198-353	3385	9.58	**9.15**
BAHC1	Q9P281	1357-1482	9092	**9.76**	11.13
GOG6B	A6NDN3	473-580	7947	**9.78**	11.50
NOB1	Q9ULX3	110-221	4659	**9.86**	11.93
DGKH	Q86XP1	581-705	5903	**9.86**	11.41
CASZ1	Q86V15	1589-1735	7943	**9.92**	11.38
POTED	Q86YR6	367-502	15076	**9.95**	11.48
POTEC	B2RU33	367-502	15076	**10.02**	11.43
POTEH	Q6S545	405-545	8777	**10.32**	11.90
ZKSC2	Q63HK3	586-738	18126	**10.32**	14.57
PTRF	Q6NZI2	175-297	18730	**10.35**	14.07
PERQ1	O75420	289-441	10443	**10.44**	12.35
U17L8	P0C7I0	383-530	2759	10.57	**9.99**
LRCH2	Q5VUJ6	491-642	6832	**10.66**	12.60
EMIL2	Q9BXX0	121-259	5666	**11.16**	14.16
LMO7	Q8WWI1	763-901	7893	**11.18**	12.68

S7 T-Cell Receptor Analysis

S7.1 Details

We used the 10x Genomics single-cell TCR sequencing dataset described in Section S6.1, along with the CellRanger annotations of chain features provided along with the dataset. Annotations of the reference structure PDB:2BNR are based on IgBLAST annotations (Ye et al., 2013) of the nucleotide sequence of 1G4 TCRβ obtained from Robbins et al. (2008), and translated from nucleotides into the corresponding positions in the amino acid sequence (Figure S9).

To obtain a latent space representation (Figure 5B), we trained the FactorMuE observation model with $T = 2$ latent dimensions, and chose among training runs based on a randomly held out

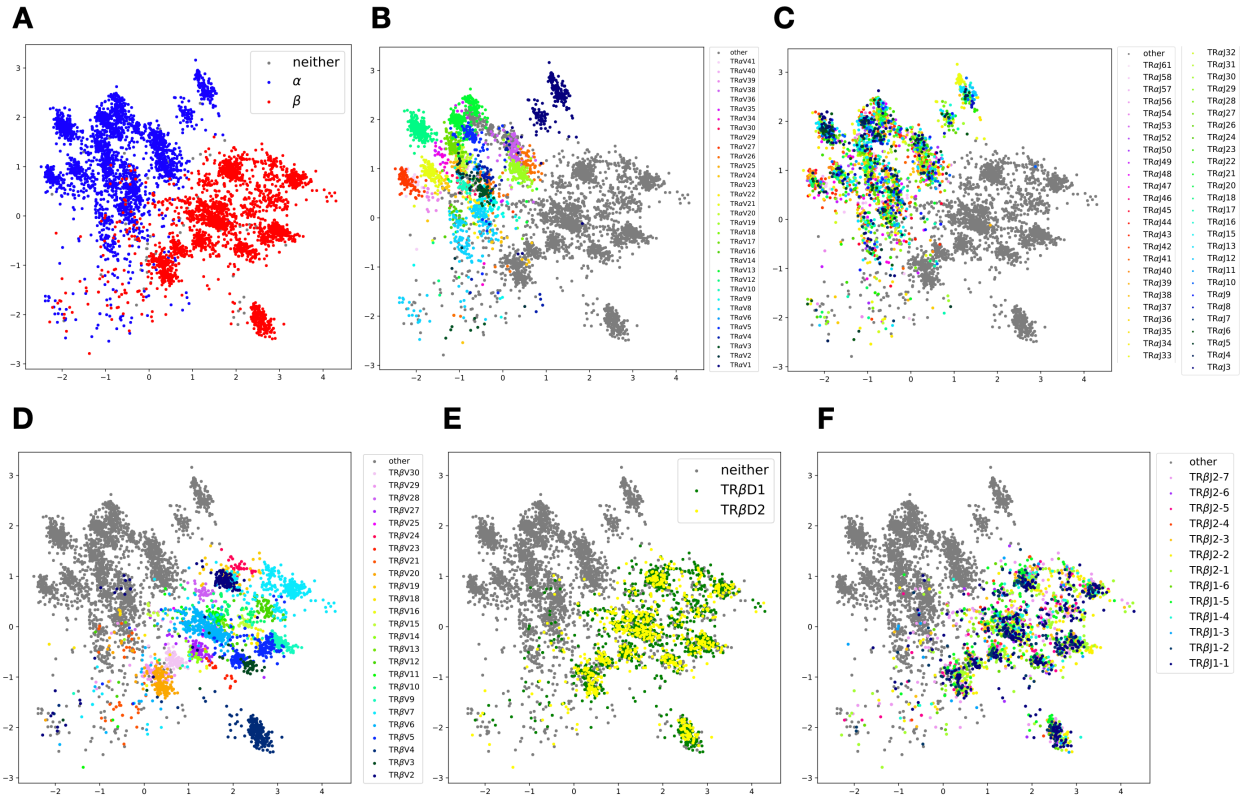

Figure S7: **Latent space representation of human T-cell receptor sequences, colored by supervised annotations.** Annotations were provided with the 10x Genomics dataset. (A) C_α versus C_β. (B) α chain V types. (C) α chain J types. (D) β chain V types. (E) β chain D types. (F) β chain J types and subtypes.

test set (5% of the data). Hyperparameters were otherwise set as in Section S6.1. The shift ν is estimated using the variational approximation to the posterior of the FactorMuE (using 10 Monte Carlo samples). $\hat{w}_{\text{ref}}$ is estimated using a single sample from the variational approximation to the posterior and the Viterbi algorithm.

S7.2 Further results

Along feature vector 2 (Figure 5D) we found weak positive correlation between the magnitude of variation and the relative surface accessibility of each site (Spearman correlation $\rho = 0.20$, $p < 0.02$; Figure S8). Along feature vector 1 (Figure 5D) we observed high values of ν_l in the V segment, suggesting that there are systematic and heterogeneous differences between the V segment sequence distribution used in TCRα chains and in TCRβ chains. To confirm the observation, we used the RegressMuE model to predict the entire TCR sequence based just on its annotation as TCRα or TCRβ. In particular, as covariate vector H_i we used a one-hot encoding of the chain type annotated by CellRanger; sequences without an annotation were labeled as $(0, 0)$. We computed the regression shift ν_l in the same way as Equation 3, with the covariate H in place of z. Figure S10 plots the shift in amino acid preference between the two chains, showing that at a population level there are key positions within the variable region with substantial differences in preference.

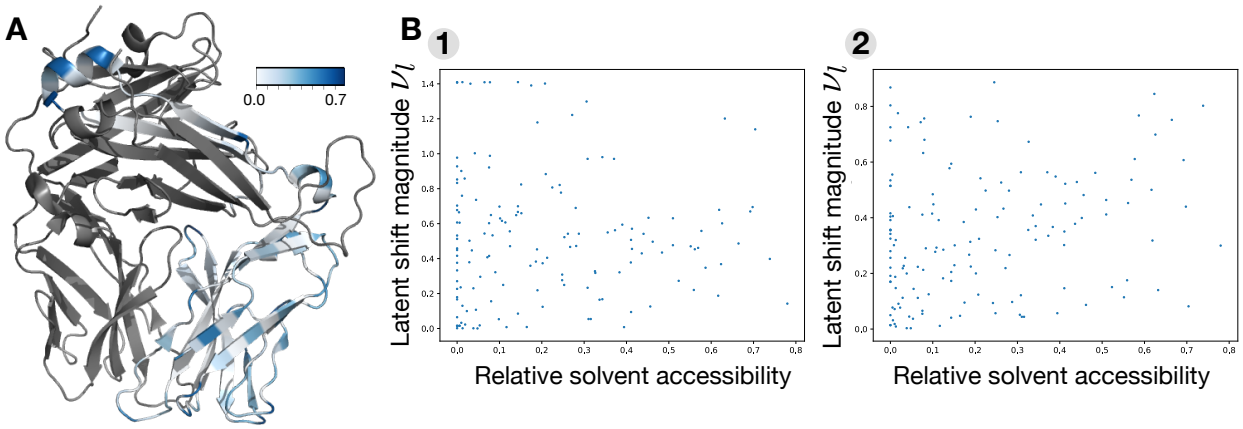

Figure S8: **Comparing MuE observation model features to T-cell receptor relative solvent accessibility.** (A) Relative solvent accessibility of TCRβ from the structure PDB:2BNR (Chen et al., 2005) (the TCRα chain is shown in gray), computed using DSSP (Kabsch and Sander, 1983) and the maximum values in Tien et al. (2013) with the Biopython API (Cock et al., 2009). (B) Residue relative solvent accessibility versus FactorMuE shift magnitude ν_l along vector 1 and vector 2 from Figure 5D. The correlation between the shift along vector 1 and the accessibility is Spearman $\rho = 0.039$, $p = 0.64$.

S8 Influenza Analysis

S8.1 Details

We downloaded publicly available influenza A(H3N2) HA sequences from GISAID (Shu and McCauley, 2017). We selected only sequences longer than 500 amino acids and with no ambiguous amino acids. Some sequences were labeled at different levels of time resolution, with annotations providing months or years rather than days; we assumed month and/or day were missing at random and imputed them uniformly at random. Following Lee et al. (2018), we randomly subsampled six sequences per month, from 1968 to October 2019, to form the dataset. In the forecasting experiments we removed the mis-annotated data identified in the 2008 outlier cluster marked by ‡ in Figure 6E prior to subsampling (GISAID identifiers `EPI_ISL_24813`, `EPI_ISL_24814`, ..., `EPI_ISL_24867`). Accession numbers for the complete dataset can be found in the Supplementary Table 1 file; our results were stable upon resampling. We extracted only the first 350 amino acids of each HA sequence, covering HA1 in the reference A(H3N2) numbering (Burke and Smith, 2014).

We used $M = 361$ in the MuE distribution. We set the prior on indels to $\mu^{(r)} = \mu^{(u)} = (1000, 1)$. We trained each model for 7 epochs, which was sufficient for convergence. Hyperparameters and training schedule were otherwise set as in Section S6.1. To produce the latent embedding in Figure 6D, however, we annealed the ELBO weighting χ only up to $\chi_0 = 0.001$ after 7 epochs, providing only very weak prior regularization such that the embedding corresponds to approximately the maximum likelihood estimator of z (and we avoid posterior collapse).

To visualize features, we trained the RegressMuE model on the full time period (1968 to 2019), with 5% of datapoints randomly held out to choose among training runs. We computed the mag-

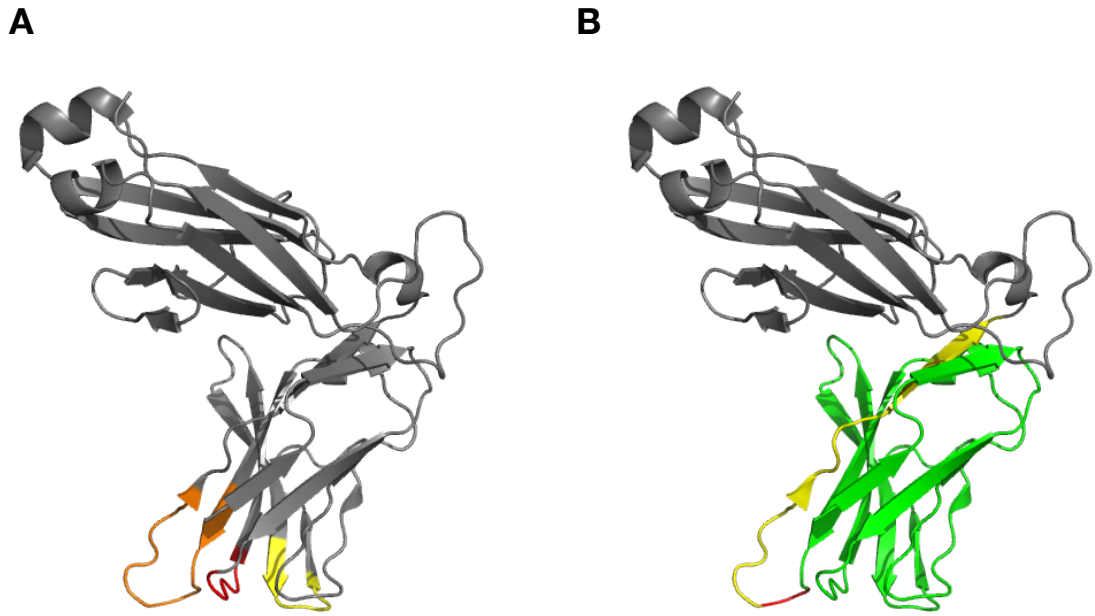

Figure S9: **T-cell receptor structural annotations.** (A) CDR segments of PDB:2BNR chain E (Chen et al., 2005), based on IgBLAST annotations (Ye et al., 2013) of the nucleotide sequence of 1G4 TCRβ obtained from Robbins et al. (2008), and translated from nucleotides into the corresponding positions in the amino acid sequence. CDR1 in red, CDR2 in yellow and CDR3 in orange. (B) V (green), J (yellow) and junction (red) segments of the 1G4 nucleotide sequence, based on the IgBLAST annotations, and translated from nucleotides.

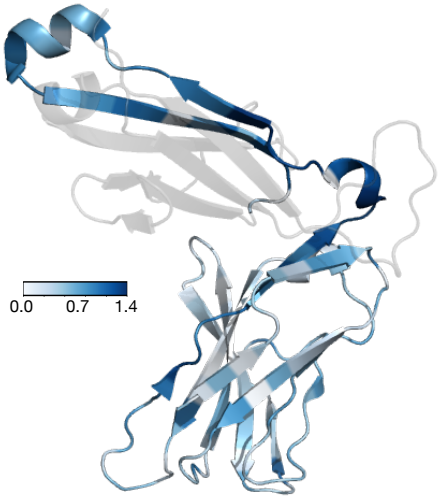

Figure S10: **Shift ν from chain α to chain β sequences learned by the RegressMuE model.** ν_l was computed as in Equation 3, using the chain annotation in place of the latent variable z.

nitude of the shift in sequence space from time t_0 to time t_1 in the RegressMuE as

$$\nu_l = \left[\sum_{b=1}^{B} \left(\mathbb{E}[Y_{l,b} | \hat{w}_{\text{ref}}, t = 2019] - \mathbb{E}[Y_{l,b} | \hat{w}_{\text{ref}}, t = 1968] \right)^2 \right]^{1/2} \tag{S67}$$

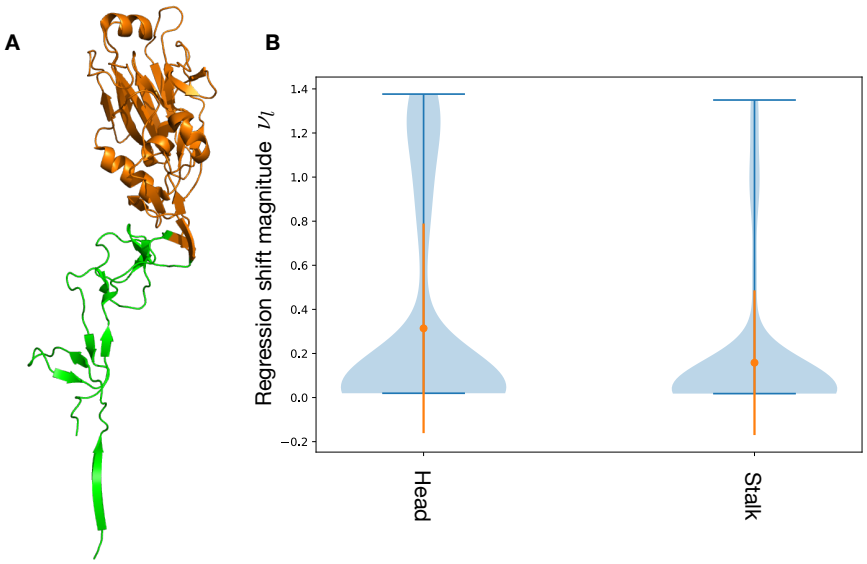

Figure S11: **Comparing RegressMuE model coefficients to HA1 structural domains.** (A) Head (orange) and stalk (green) domains of the HA1 protein (PDB:4O5N); residues between sites 52 and 277 are defined as the head domain, and all others as stalk, following Lee et al. (2018). (B) Violin plots of regression shift ν_l (Equation S67) for residues in the head domain (226 residues) versus the stalk domain (103 residues). Mean and standard deviation are shown in orange.

using as reference the HA1 sequence from PDB:4O5N. The expectation is estimated using the variational approximation to the posterior with 10 Monte Carlo samples. $\hat{w}_{\text{ref}}$ is estimated using a single sample from the variational approximation to the posterior and the Viterbi algorithm. In evaluating the association between the shift vector ν_l and epitope regions of HA1, we specifically compared to the 16 sites with clear antigenic selection in at least one human sera identified in Lee et al. (2019).

S8.2 Further results

In addition to the classic epitope regions, we also compared the regression shift ν to the structural domains of the HA1 protein (Figure S11), relative solvent accessibility (Figure S13), and relative amino acid preference in a deep mutational scan evaluating fitness effects of mutations (Figure S14).

The cluster marked ‡ in Figure 6E appears around 2008 but the latent representation of these sequences is close to that of sequences from the late 1960s or 1970s; this cluster comes from an experiment performed in 2008 on 1968 sequences, rather than contemporary patient samples as in the rest of the GISAID dataset.

MuE observation models can be used to generate samples of future sequences, enabling experimental tests of immune response and antibody titer on sequences that are likely to emerge in the future. We generated samples for the year 2024 from the RegressMuE, and confirmed that they are similar to previously observed sequences, as would be expected (Figure S15). In particular, we sampled from

$$\begin{aligned} \phi &\sim q(\phi|Y_{\mathcal{D}}) \\ Y_i &\sim p_{\text{RegressMuE}}(y|\hat{w}_{\text{ref}}, \phi, t = 2024) \end{aligned} \tag{S68}$$

where $q(\phi|Y_{\mathcal{D}})$ is the variational approximation to the posterior over model parameters, under the

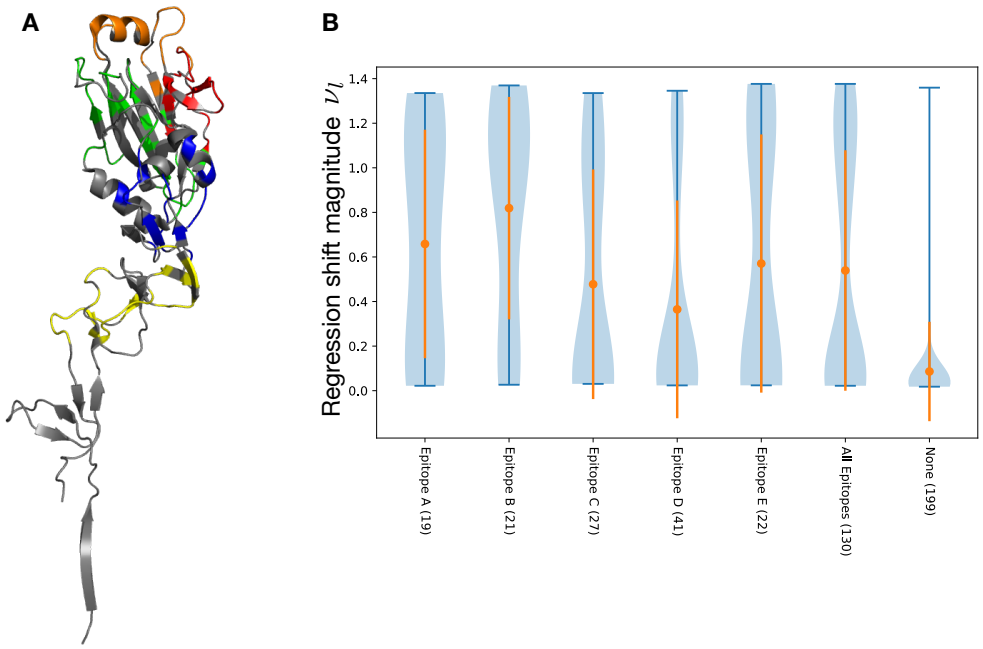

Figure S12: **Comparing MuE observation model regression coefficients to HA1 epitope regions.** (A) Epitope regions A (red), B (orange), C (yellow), D (green), E (blue) (Wiley et al., 1981; Muñoz and Deem, 2005). (B) Violin plots of regression shift ν_l (Equation S67) for residues in each epitope region, for all epitope regions together, and for residues not in any epitope region; the number of residues in each region is shown in parenthesis. Mean and standard deviation are shown in orange.

model trained on the full time period (1968 to 2019), and PDB:4O5N is again used as a reference sequence.

References

1000 Genomes Project Consortium, A. Auton, L. D. Brooks, R. M. Durbin, E. P. Garrison, H. M. Kang, J. O. Korbel, J. L. Marchini, S. McCarthy, G. A. McVean, and G. R. Abecasis. A global reference for human genetic variation. *Nature*, 526(7571):68–74, Oct. 2015.

A. A. Alemi, B. Poole, I. Fischer, J. V. Dillon, R. A. Saurous, and K. Murphy. Fixing a broken ELBO. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

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

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

S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, 2016.

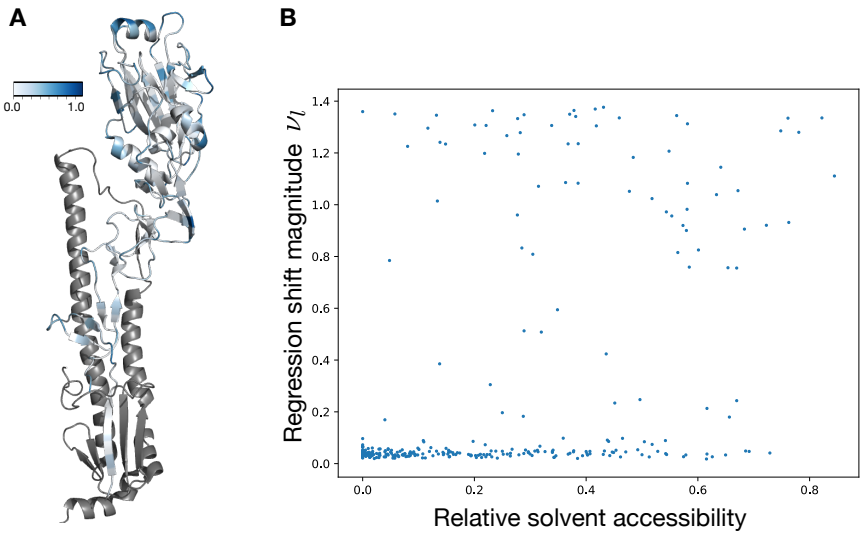

Figure S13: **Comparing MuE observation model regression coefficients to HA1 relative solvent accessibility.** (A) Relative solvent accessibility of the HA1 protein (PDB:4O5N), computed using DSSP (Kabsch and Sander, 1983) and the maximum values in Tien et al. (2013) with the Biopython API (Cock et al., 2009). HA2 protein shown in dark gray. (B) Relative solvent accessibility versus regression shift magnitude ν_l (Equation S67), residue-by-residue. Spearman $\rho = 0.41$, $p < 10^{-13}$.

D. F. Burke and D. J. Smith. A recommended numbering scheme for influenza A HA subtypes. *PLoS One*, 9(11):e112302, Nov. 2014.

J.-L. Chen, G. Stewart-Jones, G. Bossi, N. M. Lissin, L. Wooldridge, E. M. L. Choi, G. Held, P. R. Dunbar, R. M. Esnouf, M. Sami, J. M. Boulter, P. Rizkallah, C. Renner, A. Sewell, P. A. van der Merwe, B. K. Jakobsen, G. Griffiths, E. Y. Jones, and V. Cerundolo. Structural and kinetic basis for heightened immunogenicity of T cell vaccines. *J. Exp. Med.*, 201(8):1243–1255, Apr. 2005.

P. J. A. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski, and M. J. L. de Hoon. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, June 2009.

R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

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

R. C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.*, 32(5):1792–1797, Mar. 2004.

E. Gasteiger, C. Hoogland, A. Gattiker, S. Duvaud, M. R. Wilkins, R. D. Appel, and A. Bairoch. Protein identification and analysis tools on the ExPASy server. In J. M. Walker, editor, *The Proteomics Protocols Handbook*, pages 571–607. Humana Press, Totowa, NJ, 2005.

S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. U. S. A.*, 89(22):10915–10919, Nov. 1992.

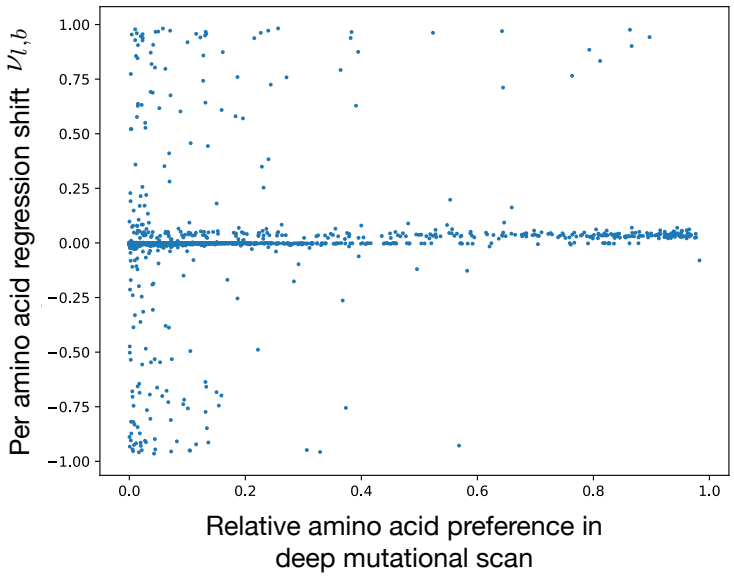

Figure S14:
Comparing MuE observation model regression coefficients to a deep mutational scan of HA. X-axis: regression shift for each amino acid at each position from 1968 to 2019,

$$\nu_{l,b} := \mathbb{E}[y_{l,b}|\hat{w}_{\text{ref}}, t = 2019] - \mathbb{E}[y_{l,b}|\hat{w}_{\text{ref}}, t = 1968]$$

(terms defined as in Equation S67). Y-axis: relative preference for point mutants with amino acid b at position l in the deep mutational scan performed in Lee et al. (2018). Spearman $\rho = 0.08$, $p < 10^{-11}$.

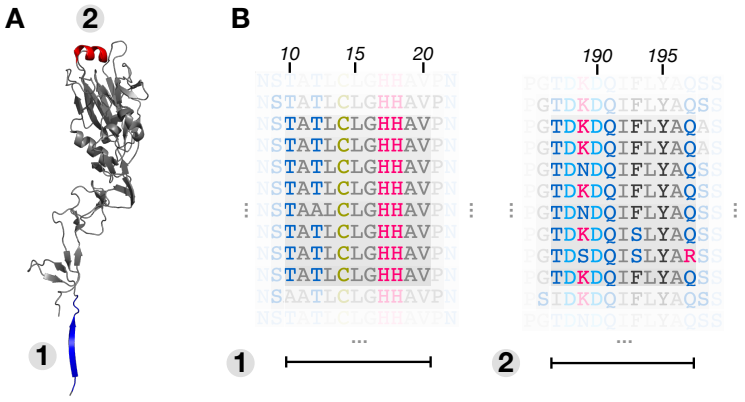

Figure S15: **Generating forecasted samples.** (A) Two locations in the reference structure PDB:4O5N, indicated in blue and red, corresponding to low and high ν_l values (Figure 6B). (B) Segments of sequences sampled from the posterior predictive distribution for the year 2024. The alignment variable w_{ref} is fixed based on the reference (PDB:4O5N), such that segments 1 and 2 correspond to the annotated structural features in A, and the column numbering is standard for influenza A(H3N2) (Burke and Smith, 2014).

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

T. A. Hopf, A. G. Green, B. Schubert, S. Mersmann, C. P. I. Schärfe, J. B. Ingraham, A. Toth-Petroczy, K. Brock, A. J. Riesselman, P. Palmedo, C. Kang, R. Sheridan, E. J. Draizen, C. Dallago, C. Sander, and D. S. Marks. The EVcouplings python framework for coevolutionary sequence analysis. *Bioinformatics*, 35(9):1582–1584, May 2019.

L. S. Johnson, S. R. Eddy, and E. Portugaly. Hidden markov model speed heuristic and iterative HMM search procedure. *BMC Bioinformatics*, 11:431, Aug. 2010.

W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, Dec. 1983.

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

D. P. Kingma and M. Welling. Auto-Encoding variational bayes. In *International Conference on Learning Representations ICLR*, Apr. 2014.

A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, and D. M. Blei. Automatic differentiation variational inference. *J. Mach. Learn. Res.*, 18(14):1–45, Jan. 2017.

J. M. Lee, J. Huddleston, M. B. Doud, K. A. Hooper, N. C. Wu, T. Bedford, and J. D. Bloom. Deep mutational scanning of hemagglutinin helps predict evolutionary fates of human H3N2 influenza variants. *Proc. Natl. Acad. Sci. U. S. A.*, 115(35):E8276–E8285, Aug. 2018.

J. M. Lee, R. Eguia, S. J. Zost, S. Choudhary, P. C. Wilson, T. Bedford, T. Stevens-Ayers, M. Boeckh, A. C. Hurt, S. S. Lakdawala, S. E. Hensley, and J. D. Bloom. Mapping person-to-person variation in viral mutations that escape polyclonal serum targeting influenza hemagglutinin. *Elife*, 8, Aug. 2019.

E. T. Muñoz and M. W. Deem. Epitope analysis for influenza vaccine design. *Vaccine*, 23(9): 1144–1148, Jan. 2005.

K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48(3):443–453, 1970.

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

R. Ranganath, S. Gerrish, and D. M. Blei. Black box variational inference. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, 2014.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

P. F. Robbins, Y. F. Li, M. El-Gamil, Y. Zhao, J. A. Wargo, Z. Zheng, H. Xu, R. A. Morgan, S. A. Feldman, L. A. Johnson, A. D. Bennett, S. M. Dunn, T. M. Mahon, B. K. Jakobsen, and S. A. Rosenberg. Single and dual amino acid substitutions in TCR CDRs can enhance antigen-specific T cell functions. *J. Immunol.*, 180(9):6116–6131, May 2008.

S. Särkkä and Á. F. García-Fernández. Temporal parallelization of bayesian smoothers. *IEEE Trans. Automat. Contr.*, 66(1):299–306, 2020.

Y. Shu and J. McCauley. GISAID: Global initiative on sharing all influenza data - from vision to reality. *Euro Surveill.*, 22(13), Mar. 2017.

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

E. R. Tamarozzi and S. Giuliatti. Understanding the role of intrinsic disorder of viral proteins in the oncogenicity of different types of HPV. *Int. J. Mol. Sci.*, 19(1), Jan. 2018.

J. L. Thorne, H. Kishino, and J. Felsenstein. An evolutionary model for maximum likelihood alignment of DNA sequences. *J. Mol. Evol.*, 33(2):114–124, Aug. 1991.

M. Z. Tien, A. G. Meyer, D. K. Sydykova, S. J. Spielman, and C. O. Wilke. Maximum allowed solvent accessibilites of residues in proteins. *PLoS One*, 8(11):e80635, Nov. 2013.

A. Toth-Petroczy, P. Palmedo, J. Ingraham, T. A. Hopf, B. Berger, C. Sander, and D. S. Marks. Structured states of disordered proteins from genomic sequences. *Cell*, 167(1):158–170.e12, Sept. 2016.

D. Tran, M. Hoffman, D. Moore, C. Suter, S. Vasudevan, A. Radul, M. Johnson, and R. A. Saurous. Simple, distributed, and accelerated probabilistic programming. In *Neural Information Processing Systems*, 2018.

UniProt Consortium. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Res.*, 47(D1): D506–D515, Jan. 2019.

S. Vogel, H. Ney, and C. Tillmann. HMM-based word alignment in statistical translation. In *Proc. of the 16th International Conference on Computational Linguistics (COLING '96)*, pages 836–841, 1996.

D. C. Wiley, I. A. Wilson, and J. J. Skehel. Structural identification of sites of Hong Kong influenza and their involvement in antigenic variation. *Nature*, 289, 1981.

J. Ye, N. Ma, T. L. Madden, and J. M. Ostell. IgBLAST: an immunoglobulin variable domain sequence analysis tool. *Nucleic Acids Res.*, 41(Web Server issue):W34–40, July 2013.