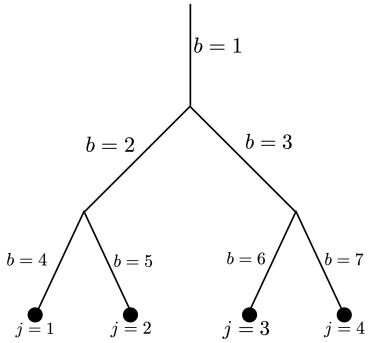

Figure 5: Tree labeling for the proof of Proposition 2.3

A Societal Impact Statement

Fitness estimation methods have wide potential for positive impact, including in understanding and diagnosing disease [19] and designing novel proteins that can accelerate research or treat disease [50]. However, like most powerful technologies, fitness estimation methods also have potential for negative impact. Improved ability to diagnose genetic disease could enable discrimination on the basis of genetic mutations, for instance insurance companies could refuse coverage or charge high premiums to patients with pathogenic variants. These risks can be mitigated via appropriate laws and regulations, such as the Genetic Information Nondiscrimination Act of 2008 in the US. Synthetic biology and molecular design are dual use technologies, and novel proteins could be designed that cause harm. For a further discussion of dual use concerns for machine learning-based molecular design, and recommendations for mitigation, see Urbina et al. [54].

B Evolutionary dynamics models

Application of the Sella and Hirsh [49] model (Eqn. 1) in JFPMs rests on a number of assumptions; we briefly the most relevant here.

When applying Eqn. 1 to amino acid sequences, as is typical for fitness estimation models, we ignore biases that come from the genetic code, which can modify the steady state probability of amino acids (in the absence of fitness effects) away from a uniform distribution. This is justified practically by the small effect sizes: if at steady state an amino acid has probability $1/64$ instead of $1/20$, the total difference in log probability is $\log(1/20) - \log(1/64) \approx 1$, which is small compared to (for instance) the log probability differences relevant for disease risk prediction with fitness models, which are ≈ 10 ([19], Extended data Fig. 3). Moreover, this bias only contributes an overall shift in amino acid probabilities, independent of position, and so does not change our main theoretical results. We ignore biases caused by asymmetric mutation rates for analogous reasons (though note they are often included in PMs in practice) [49].

The constant β depends on the effective population size, as well as the underlying population genetics model (Moran or Wright) and organismal ploidy ([49], Table 1). Following standard practice, we treat β as fixed for simplicity, though in reality it may vary over time and across lineages. Taking into account these possible changes clearly would not contradict our main theoretical result, that fitness and phylogeny are non-identifiable.

C Proofs

C.1 Proof of Proposition 2.3

N.b. this result is known in the literature (see e.g. [22], Eqn. 1) but we are unaware of a proof, so we provide one here for completeness.

Proof. For notational convenience, we will work with a standardized OUT, with $\mu = 0$ and $\sigma = 1$. The final result can be obtained by translating and scaling the distribution of leaves. The transition distribution from point x' at time t' to point X at time t under the Ornstein-Uhlenbeck (OU) process is

$$X \sim \text{Normal}\left(x' e^{-\frac{1}{2}(t-t')}, 1 - e^{-(t-t')}\right). \tag{11}$$

This distribution can be reparameterized in location-scale form as

$$\epsilon \sim \text{Normal}(0, 1)$$
$$X = x' e^{-\frac{1}{2}(t-t')} + \sqrt{1 - e^{-(t-t')}}\epsilon.$$

As $t \to \infty$ we reach the stationary distribution $\text{Normal}(0, 1)$. Let $b \in \{1, ..., \mathbf{B}\}$ index the branches of the tree, let λ_b be the length of branch b, and let $j \in \{1, ..., N\}$ index the leaves (observed species or sequences); see Fig. 5. We have assumed that the most recent common ancestor of the observed sequences was sampled from p^∞; this can be represented by adding a single branch length (indexed $b = 1$) to the root with length $\lambda_1 = \infty$. Let ϵ_b be the noise describing the OU diffusion over each branch. Let $\xi_{j,b}$ be the total time from leaf j to the nearest vertex on branch b, so long as branch b is on the path from leaf j to the root; otherwise, set $\xi_{j,b} = \infty$. For instance, in the diagram in Figure 5, we have $\xi_{1,4} = 0$, $\xi_{1,2} = \lambda_4$, $\xi_{1,1} = \lambda_4 + \lambda_2$, and $\xi_{1,5} = \xi_{1,6} = \xi_{1,7} = \xi_{1,3} = \infty$. We can now write the leaf position as

$$X_j = \sum_b e^{-\frac{1}{2}\xi_{j,b}} \sqrt{1 - e^{-\lambda_b}} \epsilon_b. \tag{12}$$

Define the matrix

$$M_{j,b} = e^{-\frac{1}{2}\xi_{j,b}} \sqrt{1 - e^{-\lambda_b}}, \tag{13}$$

such that $X_j = \sum_b M_{j,b}\epsilon_b$. We can now describe the complete leaf distribution as

$$\vec{\epsilon} \sim \text{MultivariateNormal}(0, I_{\mathbf{B}})$$
$$X_{1:N} = M \cdot \vec{\epsilon},$$

where $I_{\mathbf{B}}$ is the $\mathbf{B}$-dimensional identity matrix. Thus, according to the location-scale representation of the multivariate normal,

$$X_{1:N} \sim \text{MultivariateNormal}(0, MM^\top). \tag{14}$$

We can simplify the covariance matrix $\Sigma := MM^\top$. First

$$\Sigma_{j,j'} = \sum_b M_{j,b} M_{j',b} = \sum_b e^{-\frac{1}{2}(\xi_{j,b} + \xi_{j',b})}(1 - e^{-\lambda_b}).$$

Before introducing the notation required to derive the general result, it's helpful to get a sense of how the derivation works; in the example tree (Figure 5),

$$\begin{aligned}
\Sigma_{1,2} &= e^{-\frac{1}{2}(\lambda_4+\lambda_5)}(1 - e^{-\lambda_2}) + e^{-\frac{1}{2}(\lambda_4+\lambda_5+2\lambda_2)}(1 - e^{-\lambda_1}) \\
&= e^{-\frac{1}{2}(\lambda_4+\lambda_5)} + (-e^{-\frac{1}{2}(\lambda_4+\lambda_5+2\lambda_2)} + e^{-\frac{1}{2}(\lambda_4+\lambda_5+2\lambda_2)}) - e^{-\frac{1}{2}(\lambda_4+\lambda_5+2\lambda_2-2\lambda_1)} \\
&= e^{-\frac{1}{2}(\lambda_4+\lambda_5)}.
\end{aligned}$$

The sum over b telescopes, leaving only the initial term, which corresponds to the total time between leaf node 1 and leaf node 2. To construct the general result, define $\tilde{b}_{j,j'}$ as the branch whose later node is the most recent common ancestor of leaves j and j'. In the example in Figure 6, $\tilde{b}_{2,4} = 4$. Let R be an ordered list of branches from $\tilde{b}_{j,j'}$ to $b = 1$, the earliest branch. In the example in Figure 6, $R = [4, 2, 1]$. Finally, let $t_{jj'}$ be the length of the shortest path from leaf j to leaf j', the time from the most recent common ancestor to j plus the time to j'. In the example in Figure 6, $t_{2,4} = \lambda_5 + \lambda_6 + \lambda_8$. We now have

$$\begin{aligned}
\Sigma_{jj'} &= \sum_{b=1}^{B} e^{-\frac{1}{2}(\xi_{j,b} + \xi_{j',b})}(1 - e^{-\lambda_b}) \\
&= \sum_{b \in R} e^{-\frac{1}{2}(\xi_{j,b} + \xi_{j',b})}(1 - e^{-\lambda_b}) \\
&= e^{-\frac{1}{2}t_{jj'}} - e^{-\frac{1}{2}(t_{jj'} + 2\lambda_{\tilde{b}_{j,j'}})} + \sum_{k=2}^{|R|} e^{-\frac{1}{2}(t_{jj'} + 2\sum_{k'=1}^{k-1} \lambda_{R_{k'}})}(1 - e^{-\lambda_{R_k}}).
\end{aligned}$$

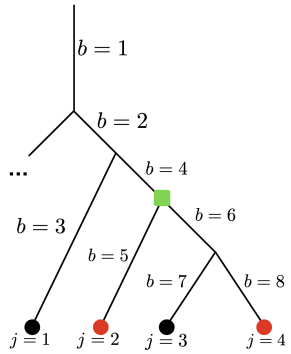

Figure 6: In red are the leaves considered in the examples in the proof of Proposition 2.3; in green is their most recent common ancestor.

Breaking down the telescoping sum, and using the fact that the final element of R is $t_1 = \infty$,

$$= e^{-\frac{1}{2}t_{jj'}} - e^{-\frac{1}{2}(t_{jj'}+2\sum_{k'=1}^{|R|}\lambda_{R_{k'}})} = e^{-\frac{1}{2}t_{jj'}}.$$

So we have the simple result that the covariance matrix depends just on the divergence times between leaves,

$$\Sigma_{jj'} = e^{-\frac{1}{2}t_{jj'}}. \tag{15}$$

Translating the distribution Eqn. 14 by μ and scaling by σ yields the result. □

C.2 Proof of Theorem 3.3

Before proving the result, we briefly clarify a definition in the statement of the theorem:

Definition C.1 (Exchangeable in leaves)**.** *Let $\mathbf{H}$ be a tree with countably infinite leaves and let $\mathbf{H}_\pi$ be a permutation of a phylogeny in its leaves, i.e. the same tree $\mathbf{H}$ with the leaves observed in a different order, according to a permutation π. A distribution over phylogenies is exchangeable in its leaves if $p(\mathbf{H}) = p(\mathbf{H}_\pi)$ for any permutation π.*

Proof. Outline: First, using the results from Sarkar [48], we construct an embedding for each tree into the hyperbolic plane, being careful that the embedding preserves exchangeability. Second, we apply de Finetti's Theorem to obtain the conditionally independent representation of the joint distribution of $Z_1, Z_2, ...$ Third, we use the distortion bound from Sarkar [48] to bound the Wasserstein distance between $p(\nu)$ and $p(\tilde{\nu})$.

First we describe the Sarkar [48] $(1+\epsilon)$ distortion embedding algorithm setup. Vertices in phylogenetic trees have maximum degree three, and, by assumption, the minimum edge length in a tree $\mathbf{H}$ is greater than $\eta > 0$ with probability one. For any $\epsilon' > 0$, choose a $\rho < \pi/3$ and a scale factor

$$\lambda > \left(\frac{1+\epsilon'}{\epsilon'}\right)\frac{2k}{\eta}\log\tan\frac{\rho}{2}, \tag{16}$$

where k is the Gaussian curvature of the hyperbolic plane $\mathbb{H}$ (for most hyperbolic geometry models, and in particular the Lorentz manifold, $k = -1$). Then, let $h_1(\mathbf{H}), h_2(\mathbf{H}), ...$ be the position of the leaves in the embedding of $\mathbf{H}$ produced by the $(1+\epsilon)$ distortion embedding algorithm in Sarkar [48], using edge scale factor λ, and ρ separated cones with cone angle $2\pi/3 - 2\rho$. Taking the last line of the proof of Theorem 6 in Sarkar [48], we are guaranteed that even for a countably infinite number of leaves,

$$\begin{aligned}\max_{i,i'}\frac{\lambda t_{ii'}(\mathbf{H})}{\tilde{d}(h_i(\mathbf{H}), h_{i'}(\mathbf{H}))} &\leq 1+\epsilon' \\ \max_{i,i'}\frac{\tilde{d}(h_i(\mathbf{H}), h_{i'}(\mathbf{H}))}{\lambda t_{ii'}(\mathbf{H})} &= 1,\end{aligned} \tag{17}$$

where $i, i' \in \mathbb{N} := \{1, 2, \ldots\}$, and $\tilde{d}(\cdot,\cdot)$ is the hyperbolic distance function.

Next we will modify the embedding function h to ensure that the distribution of embedded leaves is exchangeable. Let $[\mathbf{H}]$ be the set of phylogenetic trees that are equivalent to $\mathbf{H}$ up to reordering of the vertices. For each equivalence class $[\mathbf{H}]$ we choose one ordering of the vertices to be the canonical tree $\hat{\mathbf{H}}([\mathbf{H}])$, and for any tree $\mathbf{H}$ let $\pi^c(\mathbf{H})$ be the leaf permutation such that the reordered tree $\mathbf{H}_{\pi^c(\mathbf{H})} = \hat{\mathbf{H}}([\mathbf{H}])$. Now define the modified leaf embedding function $h'(\mathbf{H}) := h_{\pi(\mathbf{H})}(\mathbf{H}_{\pi^c(\mathbf{H})})$ where $\pi(\mathbf{H})$ is the inverse permutation of $\pi^c(\mathbf{H})$. Since by assumption the prior $p(\mathbf{H})$ on the phylogenetic tree is exchangeable, we can rewrite $p(\mathbf{H})$ using the induced distribution over equivalence classes $p([\mathbf{H}])$ as

$$\begin{aligned}[\mathbf{H}] &\sim p([\mathbf{H}]) \\ \pi &\sim \text{Permutation} \\ \mathbf{H} &:= \hat{\mathbf{H}}([\mathbf{H}])_\pi,\end{aligned}$$

where Permutation is the uniform distribution over all permutations of $\mathbb{N} := \{1, 2, \ldots\}$. We now define the distribution over leaf embeddings as

$$\begin{aligned}\mathbf{H} &\sim p(\mathbf{H}) \\ Z_{1:\infty} &:= h'_{1:\infty}(\mathbf{H}),\end{aligned} \tag{18}$$

which we can rewrite as

$$\begin{aligned}[\mathbf{H}] &\sim p([\mathbf{H}]) \\ \pi &\sim \text{Permutation} \\ Z_{1:\infty} &:= h_\pi(\hat{\mathbf{H}}([\mathbf{H}])).\end{aligned}$$

The distribution $p(Z_1, Z_2, ...)$ is therefore exchangeable. Applying de Finetti's Theorem [29] we have a.s.

$$\begin{aligned}G &\sim \mathcal{G} \\ Z_i &\overset{iid}{\sim} G \text{ for } i \in \{1, 2, \ldots\}\end{aligned} \tag{19}$$

where G is a random measure distributed according to a prior $\mathcal{G}$. Moreover, the embedding distortion bounds (Eqn. 17) are preserved for each $\mathbf{H}$, since

$$\begin{aligned}1 + \epsilon \geq \max_{i,i'} \frac{\lambda t_{ii'}(\hat{\mathbf{H}}([\mathbf{H}]))}{\tilde{d}(h_i(\hat{\mathbf{H}}([\mathbf{H}])), h_{i'}(\hat{\mathbf{H}}([\mathbf{H}])))} &= \max_{i,i'} \frac{\lambda t_{\pi_i \pi_{i'}}(\mathbf{H}_{\pi^c(\mathbf{H})})}{\tilde{d}(h_{\pi_i}(\mathbf{H}_{\pi^c(\mathbf{H})}), h_{\pi_{i'}}(\mathbf{H}_{\pi^c(\mathbf{H})}))} \\ &= \max_{i,i'} \frac{\lambda t_{ii'}(\mathbf{H})}{\tilde{d}(h'_i(\mathbf{H}), h'_{i'}(\mathbf{H}))},\end{aligned} \tag{20}$$

and by the same logic

$$1 = \max_{i,i'} \frac{\tilde{d}(h_i(\hat{\mathbf{H}}([\mathbf{H}])), h_{i'}(\hat{\mathbf{H}}([\mathbf{H}])))}{\lambda t_{ii'}(\hat{\mathbf{H}}([\mathbf{H}]))} = \max_{i,i'} \frac{\tilde{d}(h'_i(\mathbf{H}), h'_{i'}(\mathbf{H}))}{\lambda t_{ii'}(\mathbf{H})}. \tag{21}$$

We will now construct the Wasserstein bound. Define the joint distribution over ν and $\tilde{\nu}$,

$$\begin{aligned}\mathbf{H} &\sim p(\mathbf{H}) \\ \nu_{ii'}(\mathbf{H}) &:= \log(\frac{1}{2} t_{ii'}(\mathbf{H})) \\ \tilde{\nu}_{ii'}(\mathbf{H}) &:= \log(d(h'_i(\mathbf{H}), h'_{i'}(\mathbf{H})))\end{aligned} \tag{22}$$

where we have chosen $d(\cdot, \cdot) = \frac{1}{2\lambda}\tilde{d}(\cdot, \cdot)$. Note that the marginal distribution of ν matches its definition in the statement of the theorem, and that, applying Eqn. 18 and Eqn. 19, the marginal distribution of $\tilde{\nu}$ also matches its definition. Using the fact that log is a monotonically increasing function, Eqn. 20 gives

$$\begin{aligned}\log \sup_{i,i'} \frac{\exp(\nu_{ii'}(\mathbf{H}))}{\exp(\tilde{\nu}_{ii'}(\mathbf{H}))} &\leq \log(1 + \epsilon) \\ \sup_{i,i'} [\nu_{ii'}(\mathbf{H}) - \tilde{\nu}_{ii'}(\mathbf{H})] &\leq \epsilon,\end{aligned}$$

and similarly using the bound from Eqn. 21, $\sup_{i,i'}[\tilde{\nu}_{i,i'}(\mathbf{H}) - \nu_{i,i'}(\mathbf{H})] \leq 0$. Thus, with probability 1 under $p(\mathbf{H})$,

$$\|\nu(\mathbf{H}) - \tilde{\nu}(\mathbf{H})\|_\infty = \sup_{i,i'} |\nu_{ii'}(\mathbf{H}) - \tilde{\nu}_{ii'}(\mathbf{H})| \leq \epsilon.$$

Recall that the Wasserstein distance between the distribution of two random variables ν and $\tilde{\nu}$ can be written as

$$\mathcal{W}_1(p(\nu), p(\tilde{\nu})) = \inf_{\gamma \in \mathcal{J}} \mathbb{E}_\gamma[\|\nu - \tilde{\nu}\|_\infty]$$

where $\mathcal{J}$ is the set of joint distributions with marginals corresponding to the distributions of ν and $\tilde{\nu}$ ([15], Chap. 11.8). Using the joint distribution in Eqn. 22, the Wasserstein distance is bounded by

$$\mathcal{W}_1(p(\nu), p(\tilde{\nu})) \leq \mathbb{E}_{\mathbf{H} \sim p(\mathbf{H})}[\|\nu(\mathbf{H}) - \tilde{\nu}(\mathbf{H})\|_\infty] \leq \epsilon. \tag{23}$$

Now consider the case where $\mathcal{W}_1(p(\nu), p(\tilde{\nu})) = 0$. (N.b. in this case, we do not need to assume that the minimum time between nodes in $\mathbf{H}$ is greater than $\eta > 0$.) Since the Wasserstein metric is a metric on the space of probability distributions ([15] Lemma 11.8.3), $p(\nu) = p(\tilde{\nu})$ a.e.. Using the standard properties of Gaussian processes ([58], Chap. 2), the GPLVM model (Eqn. 5) can be written as

$$\begin{aligned} G &\sim \mathcal{G} \\ Z_i &\overset{iid}{\sim} G \text{ for } i \in \mathbb{N} \\ \tilde{\nu}_{ii'} &:= \log d(Z_i, Z_{i'}) \\ X_{1:\infty} &\sim \text{MultivariateNormal}(\mu, \Sigma_{ii'} := \sigma^2 \exp(-\exp \tilde{\nu}_{ii'})), \end{aligned} \tag{24}$$

which is equivalent to the OUT distribution,

$$\begin{aligned} \mathbf{H} &\sim p(\mathbf{H}) \\ \nu_{ii'} &:= \log[\frac{1}{2} t_{ii'}(\mathbf{H})] \\ X'_{1:\infty} &\sim \text{MultivariateNormal}(\mu, \Sigma_{i,i'} := \sigma^2 \exp(-\exp \nu_{i,i'})). \end{aligned} \tag{25}$$

So the distribution $p(X_{1:\infty})$ produced by the GPLVM is equivalent to the distribution $p(X'_{1:\infty})$ produced by the OUT model a.e.. □

C.3 Proof of Theorem 4.1

We start with a more basic result that captures the intuition behind Thm. 4.1, and then prove a more general result, from which Thm. 4.1 can be derived as a corollary. In particular, we start by examining the special case where the model is well-specified with respect to p^∞.

Proposition C.2. *Assume that the model $\mathcal{M}$ is log-convex and well-specified with respect to the stationary distribution, i.e. $p^\infty \in \mathcal{M}$. Assume q_{θ^*} exists and is unique. Then, if the model is misspecified with respect to the data distribution, i.e. $p_0 \notin \mathcal{M}$, we have*

$$\text{KL}(q_{\theta^*} \| p^\infty) < \text{KL}(p_0 \| q_{\theta^*}) + \text{KL}(q_{\theta^*} \| p^\infty) \leq \text{KL}(p_0 \| p^\infty). \tag{26}$$

But if the model is well-specified, i.e. $p_0 \in \mathcal{M}$, we have

$$\text{KL}(q_{\theta^*} \| p^\infty) = \text{KL}(p_0 \| p^\infty). \tag{27}$$

Proof. The first inequality in Eqn. 26 follows from the fact that $\mathcal{M}$ is misspecified with respect to p_0 and so $\text{KL}(p_0 \| q_{\theta^*}) > 0$. The second inequality follows from Thm. 1 from Csiszar and Matus [11], who study the geometry of reverse I-projections. For Eqn. 27, note that $q_{\theta^*} = p_0$ when $p_0 \in \mathcal{M}$. □

We next extend Prop. C.2 to the case where the model may be misspecified with respect to p^∞ as well as p_0.

Proposition C.3. *Assume the model $\mathcal{M}$ is log-convex and that q_{θ^*} exists and is unique. If*

$$\min_{q_\theta \in \mathcal{M}} \left(\mathbb{E}_{X \sim p_0} \left[\log \frac{p^\infty(X)}{q_\theta(X)} \right] + \mathbb{E}_{X \sim q_{\theta^*}} \left[\log \frac{q_\theta(X)}{p^\infty(X)} \right] \right) < \text{KL}(p_0 \| q_{\theta^*}) \tag{28}$$

then $\text{KL}(q_{\theta^*} \| p^\infty) < \text{KL}(p_0 \| p^\infty)$.

Proof. Define the projected stationary distribution

$$p_*^\infty = \text{argmin}_{q_\theta \in \mathcal{M}} \left(\mathbb{E}_{X\sim p_0} \left[\log \frac{p^\infty(X)}{q_\theta(X)} \right] + \mathbb{E}_{X\sim q_{\theta^*}} \left[\log \frac{q_\theta(X)}{p^\infty(X)} \right] \right)$$

Now, from the definition of the KL divergence we have

$$\text{KL}(p_0 \| p^\infty) = \text{KL}(p_0 \| p_*^\infty) + \mathbb{E}_{X\sim p_0} \left[\log \frac{p_*^\infty(X)}{p^\infty(X)} \right].$$

Applying Thm.1 from Csiszar and Matus [11] to $\text{KL}(p_0 \| p_*^\infty)$ we have

$$\begin{aligned} \text{KL}(p_0 \| p^\infty) &\geq \text{KL}(p_0 \| q_{\theta^*}) + \text{KL}(q_{\theta^*} \| p_*^\infty) + \mathbb{E}_{X\sim p_0} \left[\log \frac{p_*^\infty(x)}{p^\infty(x)} \right] \\ &= \text{KL}(p_0 \| q_{\theta^*}) + \text{KL}(q_{\theta^*} \| p^\infty) + \mathbb{E}_{X\sim q_{\theta^*}} \left[\log \frac{p^\infty(x)}{p_*^\infty(x)} \right] + \mathbb{E}_{X\sim p_0} \left[\log \frac{p_*^\infty(X)}{p^\infty(x)} \right] \end{aligned}$$

Applying Eqn. 28, the result follows. □

One way of satisfying Prop. C.3 is for there to exist a $p_*^\infty \in \mathcal{M}$ that is very close to p^∞, in the sense that $\log p^\infty(x)/p_*^\infty(x) \approx 0$ in areas of $\mathcal{X}$ with high probability under both p_0 and q_{θ^*}.

Finally we, derive the more interpretable (but also more restrictive) conditions in Thm. 4.1.
Proof of Thm. 4.1 For Eqn. 8, note that $q_{\theta^*} = p_0$ when $p_0 \in \mathcal{M}$. To show Eqn. 7, we will show that the conditions of Prop. C.3 are met. (N.b. we will work with sums over x since we are concerned with discrete sequences, but the same derivation holds replacing sums with integrals.) We have

$$\begin{aligned} &\min_{q_\theta \in \mathcal{M}} \left(\mathbb{E}_{X\sim p_0} \left[\log \frac{p^\infty(X)}{q_\theta(X)} \right] + \mathbb{E}_{X\sim q_{\theta^*}} \left[\log \frac{q_\theta(X)}{p^\infty(X)} \right] \right) \\ &\leq \min_{q_\theta \in \mathcal{M}} \sum_{x\in\mathcal{X}} \left| p_0(x) - q_{\theta^*}(x) \right| \left| \log \frac{p^\infty(x)}{q_\theta(x)} \right| \\ &\leq \min_{q_\theta \in \mathcal{M}} \| \log q_\theta - \log p^\infty \|_\infty \sum_{x\in\mathcal{X}} \left| p_0(x) - q_{\theta^*}(x) \right|. \end{aligned}$$

Applying Eqn. 6 to the first term and the definition of total variation distance to the second term,

$$< 2\,\text{TV}(q_{\theta^*} \| p_0)^2 \leq \text{KL}(p_0 \| q_{\theta^*}),$$

where the second inequality is Pinsker's inequality. We see that Eqn. 28 is satisfied, and the result follows. □

D Simulation Details

In both scenarios, we generated sequences of fixed length $|X| = 30$, with an alphabet size of $B + 1 = 4$ (corresponding to nucleotides).

Scenario 1 We simulated from a Potts model

$$p_{\text{POTTS}}(x) = \frac{1}{\mathcal{Z}} \exp \left(\sum_l \sum_b h_{lb} x_{lb} + \sum_l \sum_{l'>l} \sum_b \sum_{b'} e_{ll'bb'} x_{lb} x_{lb'} \right)$$

where h is the sitewise energies, e is the pairwise energies, x is a one-hot sequence encoding, l indexes sequence positions and b indexes letters. Following the simulations in Ingraham and Marks [27], which were intended to roughly match the statistics of typical real protein Potts models, we drew $h_{lb} \sim \text{InvGamma}(2, 0.8)$ and

$$\begin{aligned} A_{ll'} &= \begin{cases} 1 & \text{if } l' = l+1 \\ \text{Bernoulli}(0.1) & \text{otherwise} \end{cases} \\ B_{ll'bb'} &\sim \text{Normal}(0, 1.2) \\ e_{ll'bb'} &= A_{ll'} B_{ll'bb'}. \end{aligned}$$

The energies h and e were drawn once, and the same values used across independent simulations. We sampled from the model using a Gibbs sampler with 100 steps of burn-in and 10 parallel chains using the code from Ingraham and Marks [27] (`https://github.com/debbiemarkslab/persistent-vi`). We shuffled the resulting samples to remove autocorrelation.

Scenario 2 We used a site-wise independent fitness function:

$$f(x) = \sum_{l=1}^{30} \sum_{b} h_{lb} x_{lb},$$

with site-wise residue biases h_l, where x_l is a one-hot encoding of the letter at the l-th position of x. To generate phylogenetically correlated sequences, we sampled phylogenetic trees from a Kingman Coalescent ([3], Def. 2.1) with rate 1. Starting from a random sequence drawn from the steady state distribution at the root, we evolved the sequence simulating a Wright process in a haploid population ([49], Eqn. 3) according to the tree and fitness function. In particular, for sequences x_0, x that are one mutation away, the mutation rate is

$$\lim_{\tau \to 0} \frac{1}{\tau} P^{\tau}(x, x_0) = N_{\text{eff}} \frac{e^{2(f(x)-f(x_0))} - 1}{e^{2N_{\text{eff}}(f(x)-f(x_0))} - 1},$$

where we set the effective population size to $N_{\text{eff}} = 10000$. This stochastic process has steady state

$$p^{\infty}(x) \propto \exp\left(2(N_{\text{eff}} - 1) f(x)\right),$$

([49], Eqn. 7).

SWI model We fit the SWI model with maximum likelihood estimation.

BEAR model In these simulations, we used a vanilla BEAR model with a uniform embedded AR model (i.e. a Bayesian Markov model) for simplicity. We set the Dirichlet prior concentration to the constant $\alpha = 0.5$. Based on the theoretical analysis in Amin, Weinstein and Marks [2] (Thm. 35), we used a prior on lags of the form

$$p(L) \propto \exp(-B^L) \tag{29}$$

where B is the alphabet size (4 for nucleotides). We inferred the prior via empirical Bayes, marginalizing over the transition probabilities following the protocol in [2]. Conditional on lag L, sampling from the posterior over the BEAR model is straightforward thanks to Dirichlet-Categorical conjugancy.

Evaluation We defined $\mathcal{S}_f$ following standard protocols for fitness estimation models. In particular, we let $\mathcal{S}_f(p)$ be the Spearman correlation between $p(x)$ and $f(x)$ for $x \in \Lambda$ where Λ consists of all possible single point mutations (i.e. single letter changes) of an initial ("wild-type") sequence. The wild-type sequence was chosen as the most likely sequence under p^{∞}, computed exactly for Scenario 2 and estimated based on the 10^6 samples for Scenario 1.

To estimate model perplexity (Fig. 3C and 9B), we used $N = 10,000$ independent sequences from p_0 and computed the per-residue perplexity

$$\exp\left(-\frac{1}{\sum_{n=1}^{N} |X_n|} \sum_{n=1}^{N} \log p(X_n)\right), \tag{30}$$

where $|X_n|$ is the sequence length and $p(X_n)$ is the probability of the sequence under the model.

To estimate the KL to the fitness distribution in Scenario 2 (Fig. 3D) , we sampled $N = 10,000$ independent sequences from p^{∞}, $\{X_1, \ldots, X_N\}$ and estimated

$$\text{KL}(p^{\infty} || p) \approx H(p^{\infty}) - \frac{1}{N} \sum_{n=1}^{N} \log p(X_n),$$

where $H(p^{\infty})$ is the entropy of p^{∞}, which can be computed analytically. For BEAR, we plotted the KL to the posterior predictive, which, using Jensen's inequality can also be seen to lower bound

$$\mathbb{E}_{\Pi_{\text{BEAR}}(p|X_{\text{train}})}[\text{KL}(p^{\infty} || p)],$$

where $\Pi_{\text{BEAR}}(p|X_{\text{train}})$ is the BEAR posterior learned from the training dataset.

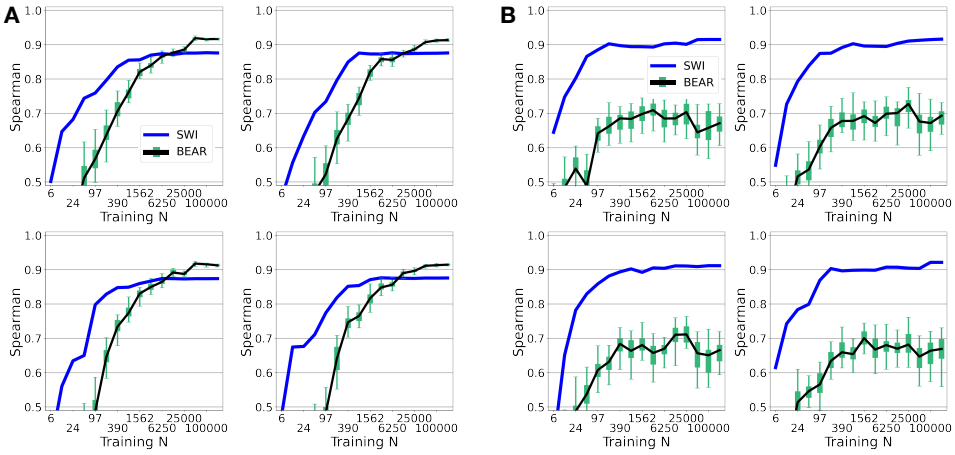

Figure 7: (A) Same as Fig. 3A, for four independent simulations following Scenario 1. (B) Same as Fig. 3B, for four independent simulations following Scenario 2.

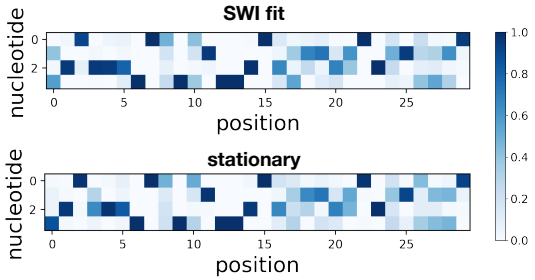

Figure 8: Probability of each nucleotide at each position learned by the SWI model (above) and in the stationary distribution p^{∞} (below), for a simulation from Scenario 2.

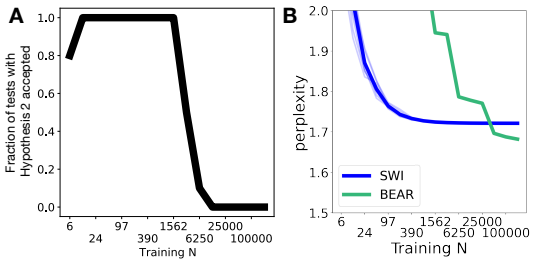

Figure 9: (A) Fraction of independent simulations (out of 10 total), following Scenario 1 (Sec. 6), in which Hypothesis 2 was accepted at level $\alpha = 0.025$. (B) Perplexity on heldout data of the BEAR and the SWI models in Scenario 1. Thick line corresponds to the average over 10 individual simulations (thin lines).

E Empirical Results Details

E.1 Data

Prediction task #1 (functional effect) Following standard practice, we report the absolute value of the Spearman correlation as $\mathcal{S}_f(p)$, since in some assays a negative change in the measured quantity corresponds to larger fitness (note that in all cases the predicted directionality of the effect under each model was correct). We focused on single amino acid substitutions, taking only those for which EVE was able to make a prediction (EVE is limited by its reliance on a multiple sequence alignment). We used the same data as in Shin et al. [50], Table 1, taking the 37 experiments performed on the following 32 proteins: UBC9_HUMAN, UBE4B_MOUSE, P84126_THETH, HIS7_YEAST, BLAT_ECOLX, IF1_ECOLI, PTEN_HUMAN, B3VI55_LIPST, GAL4_YEAST, POLG_HCVJF, PABP_YEAST, CALM1_HUMAN, AMIE_PSEAE, TRPC_THEMA, RASH_HUMAN, YAP1_HUMAN, TRPC_SULSO, DLG4_RAT, BG_STRSQ, KKA2_KLEPN, HSP82_YEAST, B3VI55_LIPST (stabilized), MK01_HUMAN, HIV BF520 env, SUMO1_HUMAN, RL401_YEAST, PA_FLU, HG_FLU, TPMT_HUMAN, HIV BG505 env, TPK1_HUMAN, and MTH3_HAEAE (stabilized). The sequence data and assay data are publicly available for research use.

Prediction task #2 (pathogenicity) We used the pathogenicity labels of single amino acid substitutions curated from ClinVar [31] in Frazer et al. [19]. Note ClinVar data is freely available for public use, and labels only depend indirectly on patient data; there is no per-person label, and no way of identifying individual patients [31]. We considered labels for 87 human proteins less than 250 amino acids in length: AICDA, AQP2, ATPF2, B9D2, CAH5A, CAV3, CD40L, CF410, CHC10, CIA30, CLD16, CLN8, COQ4, CRBB2, CRGD, CTRC, CXB1, CXB2, CXB3, CXB4, CXB6, CY24A, DERM, DGUOK, DHDDS, EDAD, EFTS, ELNE, ETFB, ETHE1, EXOS3, FGF10, FGF23, FOXE3, FRDA, GP1BB, HBB, HEM4, HSPB1, HSPB8, IFM5, IFT27, JAGN1, KAD2, KCNE1, KCNE2, KITM, LITAF, MMAB, MMAC, MPU1, MYPR, NDP, NDUS8, NFU1, NKX25, NMNA1, OPA3, PAHX, PDYN, PMM2, PMP22, PNPH, PNPO, PROP1, PSPC, PTPS, RASH, RNH2A, S5A2, SAP3, SBDS, SCO1, SDHB, SDHF2, SIX1, SIX3, SOMA, TMM70, TNNT2, TPK1, TPM2, TR13B, TWST1, VHL, XLRS1, ZC4H2.

Training data All models were trained on datasets of protein sequences gathered as described in [50] for pathogenicity effect prediction tasks and as described in [19] for functional effect prediction tasks. SWI and EVE were trained on the multiple sequence alignment, while Wavenet and BEAR were trained on the raw sequences as described in [50]. All datasets were uniformly subsampled to produce a 75%/25% train/test split.

E.2 Models and code

The SWI model was trained via maximum likelihood.

The Wavenet model was trained via maximum likelihood with the default architecture, hyperparameters and training protocol described in [50], for 100,000 steps. Code is from `https://github.com/debbiemarkslab/SeqDesign` published under the MIT licence. We did not apply the Wavenet model to the second prediction task, as it has only previously been developed for the first task.

The EVE model was trained via variational inference, using the same architecture, hyperparameters, and training protocol described in [19]. Code is from `https://github.com/debbiemarkslab/EVE` published under the MIT licence. To match the protocol of the original paper, EVE was – unlike SWI, Wavenet and BEAR – (a) trained on the full dataset rather than the training set alone, and (b) used a sequence reweighting heuristic.

The BEAR model used an embedded convolutional neural network (the same architecture as used in [2], with layer 1 width of 16, filter width of 5 and 30 filters total) and a uniform prior over lags 2, 3, 5, 7, and 9. Code is from `https://github.com/debbiemarkslab/BEAR` published under the MIT licence. The model was trained using empirical Bayes, as described in [2], for 500 steps with a batch size of 500000 kmers. Two Nvidia TeslaV100s GPUs were used, on an internal cluster; training took up to 8 hours. To construct posterior credible intervals, we used 41 samples from the posterior for prediction task #1, and 1000 samples for prediction task #2.

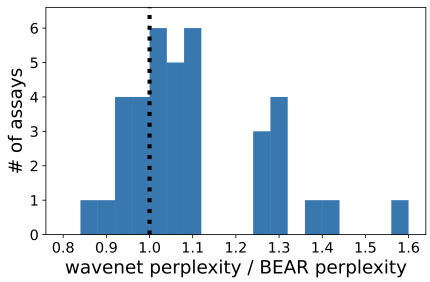

Figure 10: Ratio of the per residue perplexity on heldout data of the Wavenet model and of the BEAR model posterior predictive, across the 37 assays used for the first prediction task. Note lower perplexity corresponds to better density estimation performance.

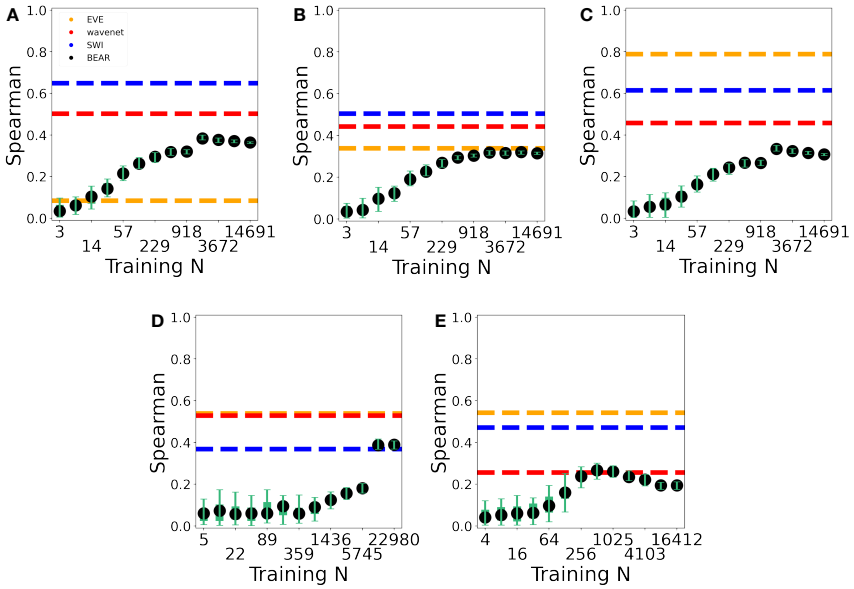

Figure 11: Same as Fig. 4CD, for 5 additional assay examples. A-C are each distinct β-lactamase assays; D is from GAL4 (DNA-binding domain); E is from UBE4B (U-box domain).

To train and predict using BEAR one needs to transform sequences into de-Bruijn graphs. To do so for amino acid sequences, we used code from `https://github.com/jdisset/kmap`. We communicated with the author of this code to obtain permission for its use.

We computed the heldout perplexity (Eqn. 30) for the BEAR posterior predictive and for Wavenet to produce Fig. 10.

E.3 Convergence experiments

To plot the convergence of the posterior over p_0 as a function of N (Fig. 4CD, 11 and 12), we used a vanilla BEAR model, a nonparametric Bayesian Markov model. Note that here we fixed the embedded AR model, rather than refitting with larger N, so that we could analyze the the convergence behavior with reference to the asymptotic results of Thm. 35 in [2], which does not take into account empirical Bayes. We set the Dirichlet concentration to 10 and used a prior over lags as in Eqn. 29.

E.4 Interpolation experiments

We fit a BEAR model using the architecture and training protocol described in Sec. E.2, optimizing both the parameters of the AR model and h via empirical Bayes. We then varied h from its optimized

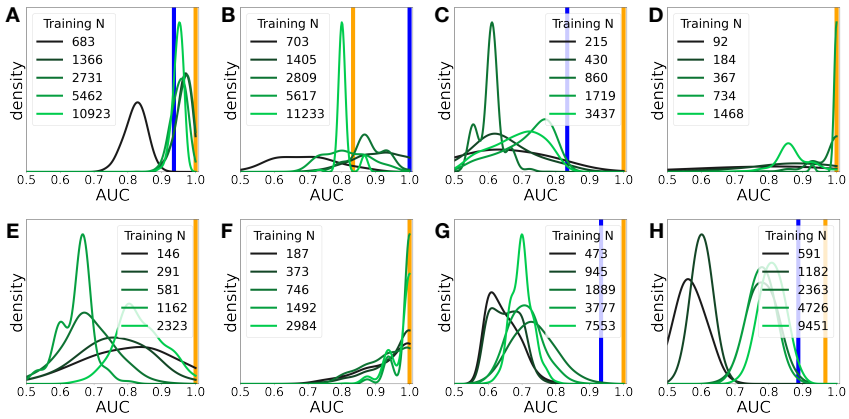

Figure 12: Convergence of the BEAR posterior over AUCs with N (green distributions), compared to the AUC of SWI (blue line) and EVE (yellow line), for the second prediction task. (A) is for the *CXB1* gene, (B) *CXB6*, (C) *EXOS3*, (D) *FGF23*, (E) *OPA3*, (F) *PAHX*, (G) *PROP1*, (H) *S5A2*.

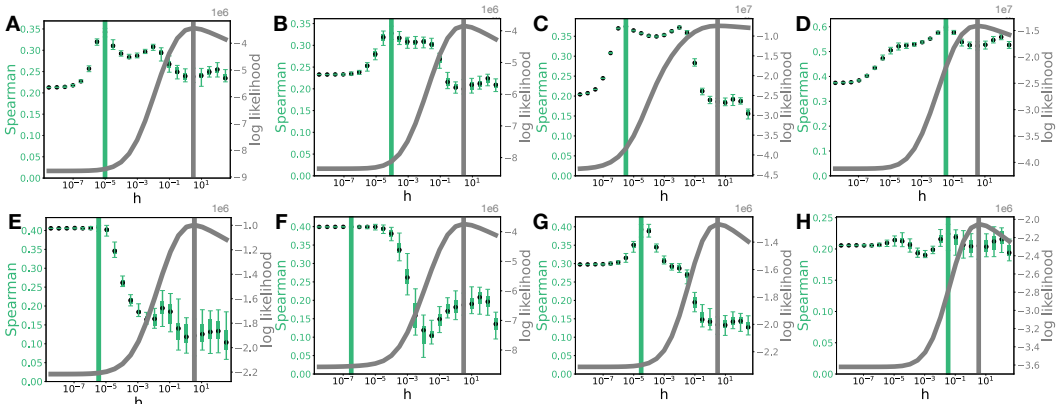

Figure 13: Same as Fig. 4EF, for 8 additional assay examples. (A) Aliphatic amidase, (B) levoglucosan kinase (stabilized), (C) HIV env protein (BF520), (D) β-glucosidase, (E) UBE4B (U-box domain) (F) TIM barrel, (G) thiopurine S-methyltransferase, (H) thiamin pyrophosphokinase 1.

value, and recalculated the total marginal likelihood and the posterior distribution over $\mathcal{S}_f(p)$ (Fig. 4EF and 13). We also computed the value of $\mathcal{S}_f(q_{\hat{\theta}})$ for the fit BEAR model in the $h \to 0$ limit (Fig. 14).

F Supplementary code

The supplementary code provides a Jupyter notebook (`example.ipynb`) illustrating the application of our BEAR diagnostic test on simulated data. It is available under an MIT License.

G Discussion of Protein Language Models

In this section we discuss the relevance and relationship of our results to large-scale "protein language models" such as ESM-1v [36], MSA Transformer [43], UniRep [1], Tranception [41], ProGen [33], ProGen2 [40] and others. Note the term "protein language model" is something of a misnomer; these methods are far from unique in applying and extending ideas from natural language processing (NLP) to build generative protein sequence models (Wavenet [50] and BEAR [2] being just two other examples). Rather, inspired by the recent success of large-scale "foundation models" in NLP, protein language models have two key distinguishing properties: (1) they use neural network architectures with very large numbers of parameters and (2) they are trained on very large and very diverse datasets, not just sequences from a single family [4].

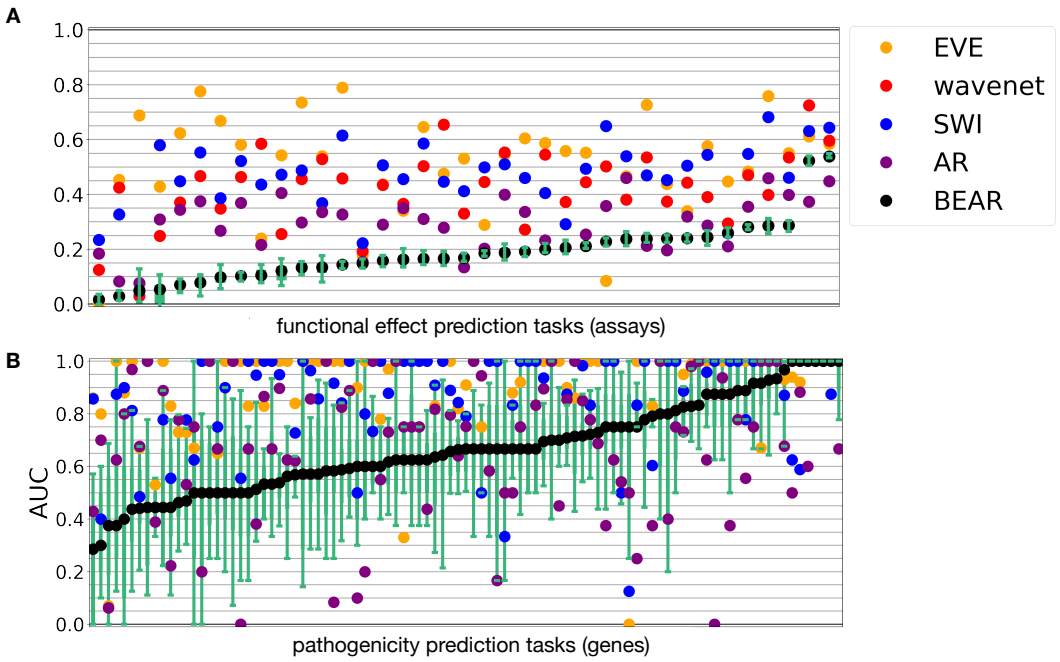

Figure 14: Same as Fig. 4AB, with the addition of the AR model in the $h \to 0$ limit (purple). In prediction task 1 (A), Hypothesis 2 is accepted in 28/37 assays (75%) while Hypothesis 1 is accepted in 6/37 (16%) for the AR model. In prediction task 2 (B), Hypothesis 2 is accepted in 16/97 genes (16%) and Hypothesis 1 is accepted in 17/97 genes (18%) for the AR model.

Our results suggest that better density estimation for sequence distributions does not necessarily imply better fitness estimation, and that a carefully chosen misspecified model can systematically outperform well-specified models. We believe our results complicate but *do not contradict* lessons learned on the importance of scale in natural and protein language models.

Big and Diverse Data Protein language models are trained on datasets that are bigger than those used for single family models, not only in the sense of containing more sequences but also in the sense of being more diverse. In other words, they differ not only in N but in the data generating distribution p_0. The idea that using such big and diverse datasets may lead to better fitness estimation is entirely in line with our theory. First, *given* a particular model class $\mathcal{M}$ and p_0, we expect datasets with larger N to be better, as it brings our estimated distribution $q_{\hat{\theta}}$ away from the prior and closer to the projected distribution q_{θ^*}. Second, while increasing the diversity of p_0 does not remove the non-identifiability problem, it can enable practical strategies for reducing the effects of phylogeny. Misspecified models fit to many different protein families may be more robust to the "noise" introduced by phylogeny, as they must be accurate across families with very different phylogenetic structure. Moreover, protein language models are often trained on datasets that have been thinned, with very similar sequences removed (e.g. UniRef90 [36] rather than UniRef100 [44]); this shifts the target distribution p_0, and can potentially ameliorate the effects of phylogeny.

Downstream Tasks The problem of fitness estimation using generative sequence models can be thought of as solving a downstream task of predicting an external label by learning an unsupervised one-dimensional representation for each sequence (namely, its log likelihood under the model).[3] In general, for problems that involve using unsupervised representations for downstream tasks, progress often comes from directly optimizing the hyperparameters of the unsupervised model to improve downstream task performance, without regard to whether such hyperparameter changes increase or decrease density estimation performance of the unsupervised model. Our results suggest that focusing on optimizing hyperparameters and architectures for downstream performance instead of density estimation, perhaps by using meta-learning, may also be a successful way forward for fitness

[3] N.b. since the representation is one-dimensional, and the downstream predictor is assumed to be a monotonically increasing function, success can be evaluated in a "zero-shot" setting using Spearman correlation or AUC, instead of training a supervised predictor on a small amount of labelled data.

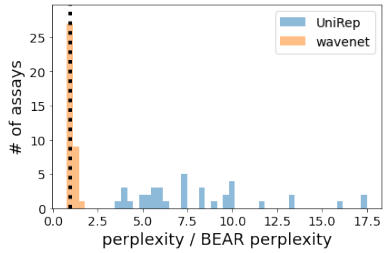

Figure 15: Ratio of the per residue perplexity on heldout data of the UniRep and Wavenet model and of the BEAR model posterior predictive, across the 37 assays used for the first prediction task. Note lower perplexity corresponds to better density estimation performance.

estimation [37].

Architecture Choices Our results are most directly in tension with recommendations from NLP for model size: we show theoretically and empirically that in the large N limit, misspecified models can be more accurate fitness estimators than well-specified models, which can conflict with the standard recommendation in large-scale natural and protein language models to give models as many parameters as possible. Our prediction is born out in the recent results of Nijkamp et al. [40], which demonstrate that a protein language model with more parameters and better density estimation can be worse at fitness estimation.

Still, a systematic understanding of model size and its tradeoffs in protein language models is lacking. For example, it may be advantageous to use misspecified models which have a very large numbers of parameters, as high-parameter models can possess other desirable properties beyond expressiveness, such as smoothness [7]. Indeed, many modern neural architectures (including attention layers in transformers) enforce invariance to certain symmetries (such as permutation or translation), ensuring that they are not universal approximators even when they have very large numbers of parameters [5]. Regularization and early stopping can also effectively restrict the model class $\mathcal{M}$.[4] Broadly, our results fit with the idea that neural architecture and hyperparameter choice can substantially impact downstream tasks even in an age of massive models and data.

The question of whether protein language models can systematically outperform the true data distribution p_0 at fitness estimation remains an open problem for future work. Here we describe some of the difficulties in addressing the problem and present preliminary results. The first challenge is scaling the diagnostic test to massive datasets. It is crucial, in particular, to work with datasets with large enough N that the test has power to reject the null hypothesis; convergence of the BEAR diagnostic is expected to be slower with (roughly speaking) high diversity p_0 (Thm. 35, [2]). Computationally, scaling the test to massive datasets will require the use of large scale amino acid kmer counters, which are currently less performant than nucleotide kmer counters [30]. A second challenge is evaluating the density estimation performance of protein language models. Models trained using masked language modeling objectives (such as ESM-1v [36] and MSA Transformer [43]) do not specify a probability distribution over full sequences, i.e. they do not estimate a well-defined density [20]. Moreover, in some cases ultra-large scale models and/or training procedures are kept private or semi-private, making it difficult to understand what exactly was in the training and test sets [33, 6].

Nevertheless, we can rule out the idea that protein language models all achieve excellent fitness estimation and excellent density estimation on single family datasets. We studied UniRep, an LSTM-based protein language model trained on a diverse evolutionary dataset (UniRef50), since UniRep provides a well-defined density over sequences (unlike ESM-1v and MSA Transformer) and is publicly available (unlike ProGen) [1]. We evaluated UniRep on the experimental assay prediction task, comparing to the BEAR posterior predictive distribution and to Wavenet, both of which were inferred from single family datasets (as described in Sec. 7). If UniRep were to show similar or better perplexity to the BEAR posterior predictive, and similar or better fitness estimation

[4] In particular, this can occur if the amount of regularization or the point of early stopping change with increasing amounts of data N, say because these values are chosen based on performance on a downstream task. In this case, regularization and early stopping will not act like a standard parametric prior, and their effects will not necessarily be asymptotically washed out in the large N limit.

performance to Wavenet, it could potentially represent a counterexample to our theory on the benefits of misspecification. Instead, we find that UniRep has substantially worse perplexity as compared to the BEAR posterior predictive (Fig. 15) and similar fitness estimation performance to the BEAR posterior predictive (median Spearman of 0.174 for UniRep, 0.169 for BEAR and 0.443 for Wavenet). We therefore see no evidence that protein language models contradict our findings on single family models, though much further work remains to be done.