Cell Systems

CellPress

Review

A new age in protein design empowered by deep learning

Hamed Khakzad,[1,2,3,5] Ilia Igashov,[2,3,5] Arne Schneuing,[2,3,5] Casper Goverde,[2,3] Michael Bronstein,[4] and Bruno Correia[2,3,*]

[1]Université de Lorraine, CNRS, Inria, LORIA, 54000 Nancy, France
[2]École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
[3]Swiss Institute of Bioinformatics (SIB), Lausanne, Switzerland
[4]University of Oxford, Oxford, UK
[5]These authors contributed equally
*Correspondence: bruno.correia@epfl.ch
https://doi.org/10.1016/j.cels.2023.10.006

SUMMARY

The rapid progress in the field of deep learning has had a significant impact on protein design. Deep learning methods have recently produced a breakthrough in protein structure prediction, leading to the availability of high-quality models for millions of proteins. Along with novel architectures for generative modeling and sequence analysis, they have revolutionized the protein design field in the past few years remarkably by improving the accuracy and ability to identify novel protein sequences and structures. Deep neural networks can now learn and extract the fundamental features of protein structures, predict how they interact with other biomolecules, and have the potential to create new effective drugs for treating disease. As their applicability in protein design is rapidly growing, we review the recent developments and technology in deep learning methods and provide examples of their performance to generate novel functional proteins.

INTRODUCTION

Proteins are the building blocks of cells and play key roles in many processes, including enzymatic reactions, transport, immune response, and signal transduction. These polypeptides are built from linear sequences of 20 different types of amino acids. The function of proteins is intrinsically related to their structure: they usually fold into a defined 3D structure known as the native conformation, though there are often other conformations the same molecule can adopt. Generally, proteins do not act in isolation but rather interact with other molecules to assemble into macro-molecular complexes. The number of natural proteins encoded in the genome is limited, and each has evolved over millions of years to perform a specific function. To further the fundamental understanding of how proteins work and to address numerous biotechnological applications, an important challenge is to *design* proteins with novel structures and functions.[1] The field of protein design has seen remarkable discoveries in the past few years, including landmark studies on designing novel enzymes,[2,3] small molecule binders,[4] protein binders,[5–7] and immunogens.[8–10] These promising achievements reveal the potential of this field and show its applicability for rational design of new macro-molecules with practical applications.

In protein design, one explores the protein space by modifying a natural sequence (directed evolution) or by designing a novel sequence from scratch (*de novo* design). In directed evolution, genetic diversity is introduced to a natural protein continued with iterative high-throughput or virtual screening and mutations to achieve a desired function mimicking natural selection.[11] In contrast, *de novo* design—the main focus of this review—does not rely on natural protein sequences and uses computational methods for designing novel proteins. Considering the protein's sequence-structure-function relationship, the three general categories of *de novo* design are depicted in Figure 1. These three main tasks include designing sequence from structure, sequence from function and structure from function. Often *de novo* design approaches aim to construct a novel protein sequence for a given protein structure. This problem setup has been referred to as "inverse protein folding."[1,12] Knowing the desired function of the target protein, however, one may explore the structure or directly the sequence space to design for function. A fundamental challenge in all these approaches is the enormous space of possible protein sequences that makes protein design computationally intensive and thus requires efficient data-driven approaches.

The interest of the protein design community in artificial intelligence (AI) and specifically in deep learning is growing rapidly as data-driven techniques promise to provide a better grasp on the immensely complex world of biology. Deep learning methods have been very successful in computer vision,[13] natural language processing (NLP),[14–16] and other applications,[17] historically considered as hallmarks of human intelligence. With the rapid growth of computational power, and high-throughput technologies allowing to produce large chemical and biological datasets, deep learning is now becoming an effective tool for biological sciences, especially in the broad field of molecular modeling and design. Early works on machine learning systems for protein design have often built on successful ideas from other domains, which have traditionally received more attention from the deep

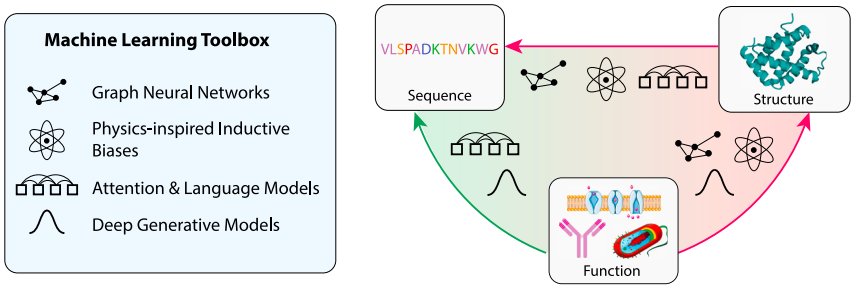

Figure 1. Three general categories of *de novo* design

The left box lists deep learning concepts that play a crucial role in modern computational protein design (see "recent innovations in deep learning" section for more detail). The diagram on the right shows how these ideas are applied in various protein design tasks. The dominant structure-to-sequence methods leverage structural information via graph representations and incorporate geometry- and physics-based inductive biases to predict relevant sequences. Function-to-structure and function-to-sequence approaches, discussed in separate sections, are mostly based on generative modeling and utilize the data representation and operations most suitable for the corresponding target data domain.

learning community. Some analogies drawn between domains are so close that transferring ideas to biological data is straightforward. For instance, NLP architectures used for machine translation can be easily applied to biological sequences because both are expressed as symbolic strings. In other cases, the biological data need to be first recast as a data structure that is amenable to the chosen deep learning model. Protein structure, for example, is often represented as a discrete voxel grid[18,19] or a set of distance and torsion angle maps,[20–23] which can be processed by convolutional neural networks (CNNs), an architecture originally developed for image processing.[17] Recently with the rapid progress of geometric deep learning methods and equivariant network architectures,[24] three-dimensional (3D) point clouds,[18] graphs,[25,26] and surfaces[27–29] have become natural and expressive means to represent proteins.

This review intends to not only provide an overview of the current capabilities of deep-learning-driven protein design but also gives application-focused researchers deeper insights into some of the most influential concepts in deep learning that help them understand and assess novel methodologies in this rapidly evolving domain. The review is structured as follow: we first describe relevant deep learning methodological building blocks underpinning most state-of-the-art protein design tools, including architectural choices, and mathematical frameworks for probabilistic modeling. This part is rather technical and is meant to be a high-level overview of deep learning methods. We then summarize the latest applications of this toolset to protein design in a way that is structured around the different ways the sequence-structure-function relationship can be inverted (see Figure 1). The review is concluded with a discussion of the salient limitations and shortcomings of state-of-the-art learning-based methods and a brief speculative outlook on potential future developments in the field. Finally, a summary of the methods covered in this review is provided in Table S1, including links to published code if available.

RECENT INNOVATIONS IN DEEP LEARNING

Before our discussion of the different ways deep learning is used by protein designers, we summarize the set of tools at their disposal. We cover neural network architectures specifically developed for processing graph-structured data, data-efficient models with carefully designed output constraints, transformers for sequential data, and generative models.

Graph neural networks

Graphs have been applied to biological problems at different levels of abstraction, from representing molecular structures to large interaction networks in cells. In protein design, graphs typically assign nodes to atoms or amino acids which are connected by edges representing chemical bonds or spatial relationships as depicted in Figure 2A. This flexibility in assigning biological units to the nodes and edges allows graph-based learning algorithms to tailor the notion of *locality* to task-specific needs. One of the main driving forces behind this shift is the rapidly growing interest in graph machine learning and message-passing neural networks[30] that has led to significant advances and superior performance on many protein design tasks.[25,26]

A *graph*, denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, is a versatile mathematical abstraction of a system of objects and their relations, where entities are represented by a set of *nodes* $\mathcal{V}$, and interactions are encoded by a set of *edges* $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Typically, the description of each node *i* is encoded in the corresponding *feature vector* $\boldsymbol{h}_i$. For instance, feature vectors of nodes in molecular graphs often consist of *one-hot encoded* atom types and possibly their 3D spatial coordinates. A key property of graph-structured data is that in most cases the ordering of the nodes is arbitrary, and thus any operation on graphs must deal with permutation symmetry (see next section for details). Hence, in a graph neural network (GNN) layer, the features of node *i* are updated via permutation-invariant aggregation $\square$ (typically sum, mean, or max) of messages $\boldsymbol{m}_{ij}$ sent by neighboring nodes *j* and a learnable function f_θ:

$$\boldsymbol{h}_i' = \mathrm{f}_\theta\left(\boldsymbol{h}_i, \square_{j \in \mathcal{N}_j} \boldsymbol{m}_{ij}\right), \qquad \text{(Equation 1)}$$

where $\mathcal{N}_j$ is a set of neighbors of the node *i* (Figure 2B). Common choices of messages defined on edges (i,j) are based on *convolutional*[31] operations of the form $\boldsymbol{m}_{ij} = c_{ij}\psi_\theta(\boldsymbol{h}_j)$, where the weights c_{ij} are fixed, *attention*[32] $\boldsymbol{m}_{ij} = a_\theta(\boldsymbol{h}_i, \boldsymbol{h}_j)\psi_\theta(\boldsymbol{h}_j)$ and most generic *message passing*[30] $\boldsymbol{m}_{ij} = \psi_\theta(\boldsymbol{h}_j, \boldsymbol{h}_j)$. In these message functions, ψ_θ and a_θ are trainable components. After multiple updates (layers), the feature vector of every node contains information about the neighborhood of the node. Normally, this information is further passed to the additional aggregation and readout operations that aim to accumulate and retrieve the data necessary for a specific task.

Physics-inspired inductive biases

An inductive bias is a prior belief or knowledge about a problem that underlies the design of learning algorithms.[33] Fundamental

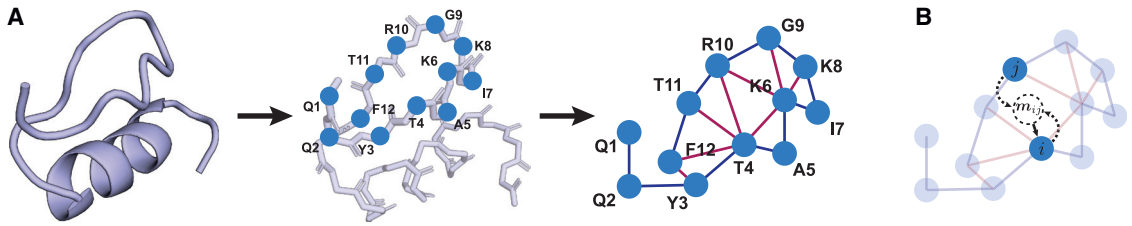

Figure 2. Example of protein residue graph

(A) Construction of the graph from a protein structure (PDB: 1FSD). Nodes are annotated with residue identities. The definition of edges is use case-specific and can be for example based on the peptide sequence (blue) or spatial proximity (magenta).

(B) Message passing. Messages are computed for all neighbors $j \in \mathcal{N}_j$ and aggregated to update the feature at node i.

inductive biases of neural networks for protein design are defined by the choice of the input representation (sequence, structure, surface, etc.) and how the data are passed to the algorithm. For example, sequence information can be encoded as a single string of amino acids, a multiple sequence alignment (MSA), or a position-specific scoring matrix (PSSM) while options for structural information include pairwise distance maps, discrete voxel grids, graphs with atomic resolution, or coarse-grained graphs. Another example is the surface representation adopted by the MaSIF framework[7,27,28] to account for the fact that interactions between proteins involve mostly their surfaces. By abstracting details about the protein cores, MaSIF achieved state-of-the-art results on molecular recognition tasks, such as protein-protein interaction prediction. The choice of the data representation strongly affects the predictions an algorithm will produce and must be carefully selected for a given task. Physics- and geometry-inspired inductive biases can help to improve data efficiency and generalization capabilities of neural networks[34] and have therefore fueled many of the recent successes of deep learning for structural biology.[25,26,35–37]

When represented as 3D objects, molecules are embedded in Euclidean space and most of their properties either do not change under translations, rotations, and reflections of this space (e.g., total energy) or transform in a consistent way (e.g., momentum). Similarly, it is not desirable for signals on graphs to depend on permutations of the nodes to avoid imposing an arbitrary order on the nodes. More generally, functions acting on signals defined on a domain with known symmetries should always produce outputs that respect these symmetries. In these cases, it is desirable to develop learnable functions that are *invariant* or *equivariant* and respect these symmetries by design. Defining such functions is a challenging task that highly depends on the data representation and the problem being addressed. We are next introducing the concepts of equivariance and invariance more formally.

A function $f : \mathcal{X} \to \mathcal{Y}$ is said to be *equivariant* with respect to the group G (e.g., the Euclidean group or the finite symmetric group in the examples above), if:

$$f\left(T_g^{\mathcal{X}}(\mathbf{x})\right) = T_g^{\mathcal{Y}}(f(\mathbf{x})) \forall \mathbf{x} \in \mathcal{X}, \forall g \in G, \quad \text{(Equation 2)}$$

where $T_g^{\mathcal{X}}$ and $T_g^{\mathcal{Y}}$ are representations of the group action $g \in G$ in the input and output domain, respectively. That is, the group action g and function f commute. In a particular case, when $T_g^{\mathcal{Y}}$ is the identity map for any $\boldsymbol{g} \in G$, the function f is said to be *invariant* with respect to the group G:

$$f\left(T_g^{\mathcal{X}}(\mathbf{x})\right) = f(\mathbf{x}) \forall \mathbf{x} \in \mathcal{X}. \quad \text{(Equation 3)}$$

These concepts are described more rigorously and in more detail in the "Geometric Deep Learning" book by Bronstein et al.[24]

An example of a rotation and translation invariant architecture applied to molecular systems is SchNet.[38] It predicts the energy of a molecule, which does not depend on rotations of the input, and guarantees energy conservation of a derived force field by differentiating the predicted total energy with respect to the atom positions. Another example is the invariant point attention (IPA) module proposed as a part of the protein structure prediction model AlphaFold2.[39] It updates all residue activations with a geometry-aware attention mechanism (see next section) using 3D queries, keys, and values in each residue's local frame, making it invariant to rigid transformations of the global reference frame.

Language models, transformers, and attention

In the past decades, new sequencing technologies enabled an exponential growth of protein sequence databases with the total number of sequences doubling up every year,[40] as shown in Figure 3A. Over the time, sequencing is becoming cheaper and the protein sequence databases grow even faster than computational power.[41] Therefore, leveraging sequential information for studying protein structural and functional properties has become a promising option especially with the outstanding success of deep learning methods designed for NLP tasks.[15,16,42] In particular, the vast amount of available protein sequences made it possible to capture statistical patterns in the proteome through training of *protein language models* (PLMs),[43,44] which aim to learn the probability distribution over the observed sequences of amino acids. Trained on a large corpus of protein sequences, protein language models are able to learn rules of protein sequences and evolution, that are often referred to as the "language of life."[41,43,44]

Unlike supervised learning methods that usually require manual annotation of the training data, protein language models are typically trained in a *self-supervised* fashion that does not require labeled datasets and therefore can use far larger amounts of data. Self-supervised methods are optimized on auxiliary objectives that can be easily defined on the unlabeled data. In the case of protein sequences, given all previous residues, a language model can be trained to predict the next amino acid in a sequence or, as Figure 3B shows, amino acids that were masked from their context (surrounding residues). Trained in

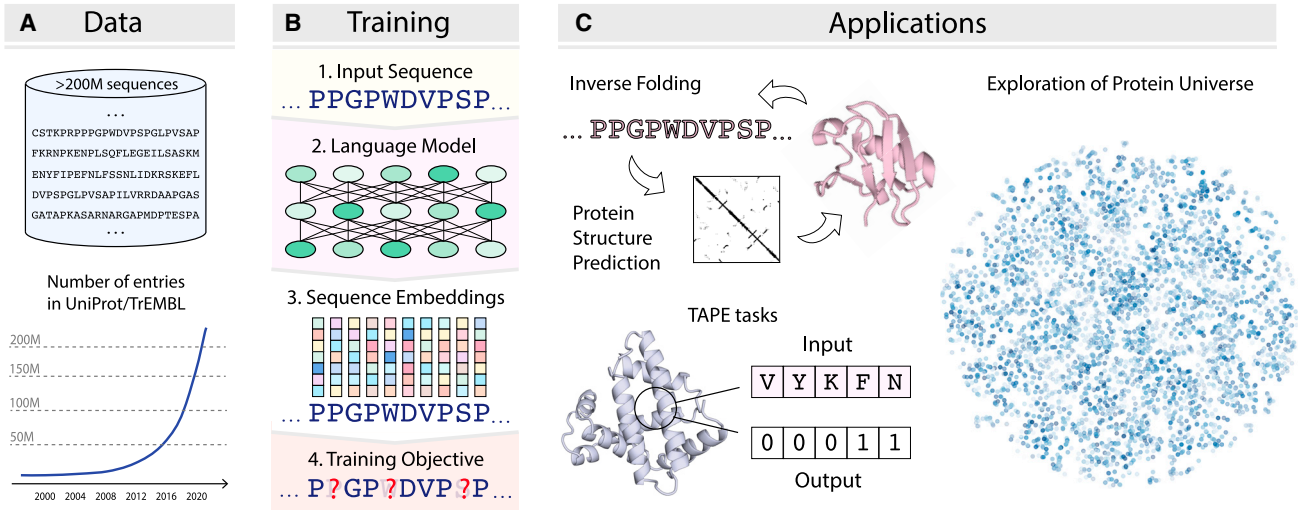

Figure 3. Protein language models

(A) Protein language models are trained on sequential data that currently exceeds 200 million sequences.[40]

(B) Self-supervised training of a language model with the masked token prediction objective.

(C) Protein language models can be applied to the inverse folding problem, protein structure prediction, and in various other tasks including tasks for assessing protein embeddings (TAPE) proposed by Rao et al.[45] Besides, sequence embeddings produced by pre-trained language models can help to explore the "protein universe."

such a task-agnostic way, protein language models can be further fine-tuned and applied in various downstream supervised and unsupervised tasks such as inverse folding and protein structure prediction, as depicted in Figure 3C.

One of the core concepts behind language models is the *transformer* architecture. It was originally proposed in the scope of the machine translation problem[14] as a more efficient alternative to recurrent neural networks (RNNs)[46–48] that have been extensively used in NLP beforehand. Among other advantages over RNNs, transformers effectively learn long-range dependencies between words in a sentence, which allows to take into account important subtle context when processing the sentence. The core transformer operation is called *attention*. For each word in an input sentence, it computes a new representation that captures contextual dependencies between this word and all other words in the sentence.

Considering an input sequence of length N, the attention mechanism is defined as follows. First, three learnable linear transformations are applied to the sentence representation resulting in three matrices: *queries* $\boldsymbol{Q} \in \mathbb{R}^{N \times d_k}$, *keys* $\boldsymbol{K} \in \mathbb{R}^{N \times d_k}$, and *values* $\boldsymbol{V} \in \mathbb{R}^{N \times d_v}$, where d_k is the size of query and key vectors and d_v is the size of value vectors. Next, queries and keys are multiplied, scaled, and normalized with a softmax function along the second dimension resulting in an $N \times N$ matrix representing contextual dependencies between all pairs of words in the sentence. Then, this matrix is multiplied with values $\boldsymbol{V}$ resulting in a final $N \times d_v$ matrix where each word in the sentence is represented by a d_v-dimensional embedding vector:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_k}}\right)\boldsymbol{V}. \qquad \text{(Equation 4)}$$

Such an attention layer stacked with feedforward neural networks using residual connections,[13] combined with positional encoding[49] and followed by layer normalization[50] constitutes a basic building block of the transformer model.[14] Depending on the task, these blocks can be combined in various ways composing more complex systems. A successful example of combining two transformer blocks is the Evoformer of AlphaFold2.[39] In Evoformer, the two transformer blocks efficiently extract structural information from the MSA and pair representations based on axial attention. These blocks are capable of exchanging information, allowing reasoning about the protein structure given co-evolutionary and homology information.

Deep generative models

Deep generative models are a class of neural networks that approximate probability distributions, which are empirically defined through large datasets. It is useful to think of them as probabilistic frameworks rather than neural network architectures. In each of the cases outlined below, different neural network backbones (e.g., CNNs, GNNs, transformers) can be used in accordance with the task at hand. Depending on the type of generative method, data distributions can be modeled either explicitly or implicitly.[51] However, all generative algorithms are designed in a way that one can sample new data points from the learned distributions. In the context of protein design, generative models hold promise because they do not rely on the accuracy of energy functions and can thus be regarded as a complementary approach to conventional design methods. This idea is gaining popularity in structure-based design as larger datasets become available. This section reviews the most popular types of deep generative models including variational autoencoders (VAEs), generative adversarial networks (GANs), and the emergent class of *diffusion models*, which are rapidly gaining traction in the protein design community.

VAEs

VAEs introduced by Kingma and Welling[52] are a landmark in the area of generative models. The framework provides a principled

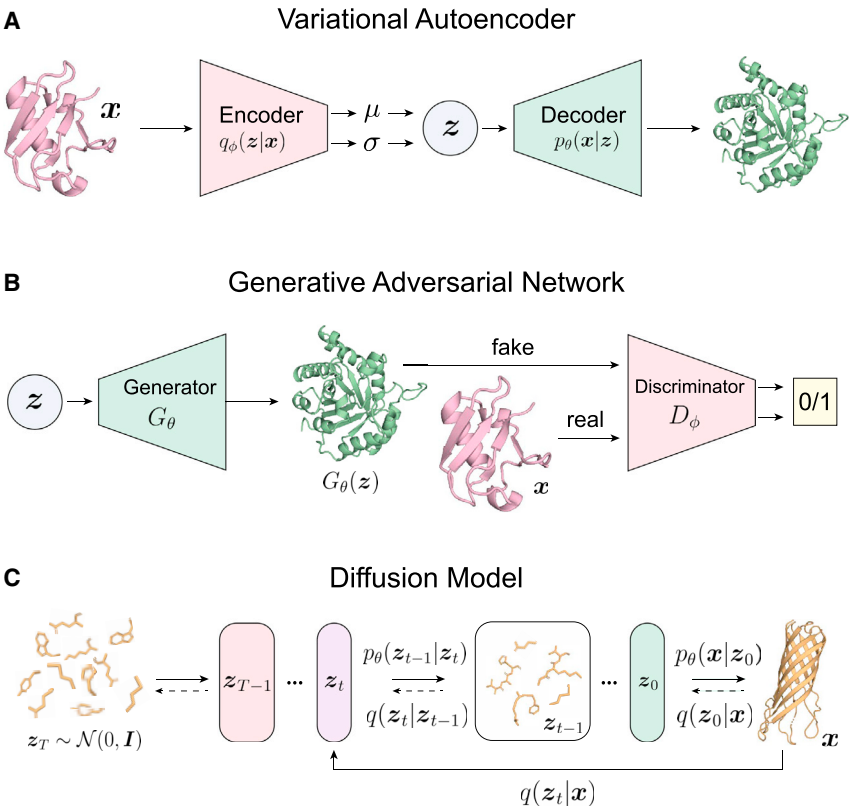

Figure 4. Deep generative models

(A) Variational autoencoders optimize a lower bound on the data likelihood $p(\boldsymbol{x})$ by jointly learning a distribution of latent variables $\boldsymbol{z}$ and a probabilistic decoder that transforms these latent vectors into data points. For generating new data, random vectors are sampled from a normal distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ in the latent space and processed by the decoder module.

(B) Generative adversarial networks consist of two components: generator $G\theta$ tries to mimic the data distribution while discriminator $D\varphi$ is trained to classify samples as "real" or "fake." After training, the generator can be used to convert random noise samples to realistic data that resemble the statistics of the training dataset.

(C) Diffusion models learn to gradually denoise perturbed versions of the training data. Typically, the original signal $\boldsymbol{x}$ is destroyed by a fixed diffusion process $q(\boldsymbol{z}_t|\boldsymbol{x})$, that turns data into noise, and the model is trained to recover data samples through a Markov chain with parameterized transition probabilities $p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$.

way to optimize weights θ of a parameterized conditional distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$, called *decoder*, that can be used to sample data points $\boldsymbol{x}$ given a latent value $\boldsymbol{z} \sim p(\boldsymbol{z})$. Since $\boldsymbol{z}$ is an unobserved random variable, its ground-truth distribution $p(\boldsymbol{z})$ is unknown and must therefore be estimated by an auxiliary model, *encoder* $q_\varphi(\boldsymbol{z}|\boldsymbol{x})$. The chained model with probabilistic encoder and decoder resembles the well-known autoencoder architecture,[53] as depicted in Figure 4A. After training, the data sampled by the model should follow the empirical distribution represented by the training set. To this end, it is natural to maximize the likelihood $p_\theta(\boldsymbol{x})$ of these data. However, the marginal likelihood $p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\mathrm{d}\boldsymbol{z}$ is often intractable and thus cannot be maximized directly. Instead, Kingma and Welling[52] proposed to optimize a lower bound on the true log-likelihood of a data point:

$$\log p_\theta(\boldsymbol{x}) \geq \mathbb{E}_{\boldsymbol{z}\sim q_\varphi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{D}_{\mathrm{KL}}(q_\varphi(\boldsymbol{z}|\boldsymbol{x})\|p(\boldsymbol{z})) \quad \text{(Equation 5)}$$

Typically, all probabilities are parameterized as Gaussians, in which case the Kullback-Leibler divergence D_{KL} can be computed in closed form to reduce the amount of Monte Carlo sampling required for estimating the variational lower bound in Equation 5. In this setup, the random variable $\boldsymbol{z}$ can be reparameterized as $\boldsymbol{z} = \boldsymbol{\mu} + \sigma\boldsymbol{\varepsilon}$ with $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, thereby decoupling the stochasticity from the deterministic encoder network.

GANs

GANs are a popular class of generative models.[54,55] The simple yet powerful idea is to pit two neural networks against each other: a *generator* $G\theta$ takes a noise vector $\boldsymbol{z}$ and transforms it into the data domain, while a *discriminator* $D\varphi$ acts as a binary classifier and tries to predict whether a given data point originates from the training set or was generated by $G\theta$, as schematically depicted in Figure 4B. Both models are parameterized as neural networks with trainable weights θ and φ, respectively. The aim of the discriminator is to assign high scores to the real data points $\boldsymbol{x}$ and to give low scores to the generated samples $G\theta(\boldsymbol{z})$. At the same time, the generator aims to produce realistic samples and therefore to maximize the scores the discriminator gives to generated samples. The overall optimization problem can be formalized as $\min_\theta \max_\varphi \mathcal{L}(\theta, \varphi)$ with the

$$\mathcal{L}(\theta, \varphi) = \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}[\log D_\varphi(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[\log(1 - D_\varphi(G_\theta(\boldsymbol{z})))], \quad \text{(Equation 6)}$$

where *pdata* is the ground-truth unknown data distribution, and $\boldsymbol{z}$ is a random noise vector typically sampled from the standard normal distribution $p(z) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Intuitively, the capabilities of both neural networks gradually improve, and finally the distribution learned by the generator is close to *pdata*.

Diffusion models

Diffusion models[56,57] are a class of generative neural networks that progressively perturb data points by adding increasing levels of noise. After a large number of steps, the original signal is completely erased resulting in a pure noise sample. Neural networks are trained to reverse this process in a stepwise manner, iteratively predicting a slightly less noisy version of the data point from its representation in the previous step. For sampling new data points, these denoising models incrementally update samples from a random noise distribution, slowly moving them onto the data manifold. This procedure is schematically shown in Figure 4C.

Diffusion models consist of two components. The first component, a *diffusion process*, progressively distorts a data point $\boldsymbol{x}$,

mapping it to noise. The probability of any intermediate state $\boldsymbol{z}_t$ for $t = 0, \ldots, T$ is commonly defined as $q(\boldsymbol{z}_t|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}_t; \alpha_t \boldsymbol{x}, \sigma_t^2 \boldsymbol{I})$, where $\alpha_t \in \mathbb{R}^+$ controls how much signal is retained and $\sigma_t \in \mathbb{R}^+$ controls how much noise is added. The reversed *true denoising* process that reconstructs the data point from noise is defined by $q(\boldsymbol{z}_{t-1}|\boldsymbol{x}, \boldsymbol{z}_t)$. It can be verified through Bayes' rule that $q(\boldsymbol{z}_{t-1}|\boldsymbol{x}, \boldsymbol{z}_t)$ is Gaussian, and its parameters can be analytically derived if the noiseless sample $\boldsymbol{x}$ is known. The second component of a diffusion model is a *generative denoising process* that learns to invert the diffusion trajectory in cases where the data point $\boldsymbol{x}$ is unknown. The generative transition distribution is defined as $p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t) = q(\boldsymbol{z}_{t-1}|\hat{\boldsymbol{x}}, \boldsymbol{z}_t)$, where $\hat{\boldsymbol{x}} = f_\theta(\boldsymbol{z}_t, t)$ is an approximation of the data point $\boldsymbol{x}$ computed by a neural network f_θ. To generate a new data point, one samples initial Gaussian noise $\boldsymbol{z}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, then intermediate states $\boldsymbol{z}_{t-1} \sim p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$ for $t = T, \ldots, 1$, and finally $\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z}_0)$. The neural network f_θ is trained by maximizing a lower bound on the log-likelihood:

$$\log p_\theta(\boldsymbol{x}) \geq \underbrace{\mathbb{E}_{q(\boldsymbol{z}_0|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z}_0)]}_{\text{Reconstruction loss}} - \underbrace{\mathrm{D}_{\mathrm{KL}}(q(\boldsymbol{z}_T|\boldsymbol{x})\|p_\theta(\boldsymbol{z}_T))}_{\text{Prior loss}} - \sum_{t=1}^{T} \underbrace{\mathrm{D}_{\mathrm{KL}}(q(\boldsymbol{z}_{t-1}|\boldsymbol{x}, \boldsymbol{z}_t)\|p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t))}_{\text{Diffusion loss}} \quad \text{(Equation 7)}$$

PROTEIN DESIGN POWERED BY DEEP LEARNING

Structure to sequence

Historically, progress in protein structure prediction has promptly been accompanied by new design tools that utilize the latest innovations. An early version of Rosetta,[58] one of the leading bioinformatics tools for protein structure prediction and design, was able to accurately predict small protein domains using combinations of structural fragments with similar local sequences and a simulated annealing scheme.[59] Shortly after, the framework was quickly equipped with a Monte Carlo optimization procedure for finding stabilizing sequences for natural folds[60] and novel structures,[61] thus transforming it into a protein design tool.[62] Similarly, the first generation of modern folding neural networks[23,63] has led to a plethora of derived design protocols,[64–66] and most recently, the unprecedented success of AlphaFold2[39] has sparked another wave of this kind.

A popular class of sequence design methods attempts to invert structure prediction models through optimization in sequence space. This procedure conceptually resembles a maximum likelihood solution for p(structure|sequence), which is approximated by the folding algorithms. A common approach is to start with a randomized amino acid sequence that will lead to a random structure and optimize it using a confidence and structural loss to a target fold. Typically, structural similarity is measured in a rotation and translation invariant way. For instance, a protein can be represented as 2D matrices of distances, angles and dihedrals for every residue pair and the differences between these inter-residue features can be used to define an optimization objective.[65] Rather than adhering strictly to the conventional approach of finding a sequence that will fold into a pre-defined target structure, protein *hallucination* explores the concept of identifying high-confidence solutions within the sequence space.[64] This moves away the focus of obtaining a predetermined structure toward searching for sequences that are highly likely to exhibit desired structural properties. The loss function is generally composed of a structural loss (constrained hallucination),[65] structural stability loss (unconstrained hallucination),[64] or a combination of both.[66] The stability loss can be derived from the confidence of the predicted structure. For example, in trRosetta,[23] the structure is predicted as an inter-residue distance map with a set of binned distances. Maximizing the probability of one of the bins using a Kullback-Leibler divergence-based loss results in a sequence and structure in which trRosetta is highly confident. Even more straightforward, AlphaFold2's confidence scores such as predicted local distance difference test (pLDDT), predicted template modeling (pTM), and predicted aligned error (pAE) can be directly plugged into optimization schemes of this kind.[67,68]

While many of the recent protocols have been developed with trRosetta[23] as their engine,[64–66] more recent works transitioned to the newer and more accurate RoseTTAfold[69] and AlphaFold2[39] structure prediction networks.[6,67,68] In addition to these dedicated structure predictors, sequence optimization is sometimes carried out with large language models equipped with a folding head (e.g., ESMFold[70] and OmegaFold[71]) or simple linear projections of the learned attention maps onto structural features.[72,73] The optimization step is usually performed in one of the two ways depicted in Figure 5. In the Markov chain Monte Carlo (MCMC) line of work,[64] new candidate sequences are sampled randomly, processed by a structure prediction model and accepted (i.e., used in the next iteration) according to a Metropolis-Hastings criterion based on the chosen loss function. This procedure is repeated until convergence of the loss function. Gradient-based approaches,[65–68] on the other hand, make use of the differentiability of deep neural networks. A structure is predicted for an initial sequence and the structural loss is calculated. Next, the error gradient to the input is back-propagated through the network and used to update a probability distribution of the amino acid identity per position using gradient descent. From this probability distribution, the most likely sequence is extracted and used for the next iteration. The algorithm typically converges faster thanks to the gradient-informed update steps. Some researchers also combined the MCMC and gradient-based strategies to benefit from the specific advantages of each method.[6,74]

Structure and stability losses have been successfully applied to a variety of protein design tasks such as free hallucination of moderately novel proteins with TM scores[75] of 0.6–0.9 over

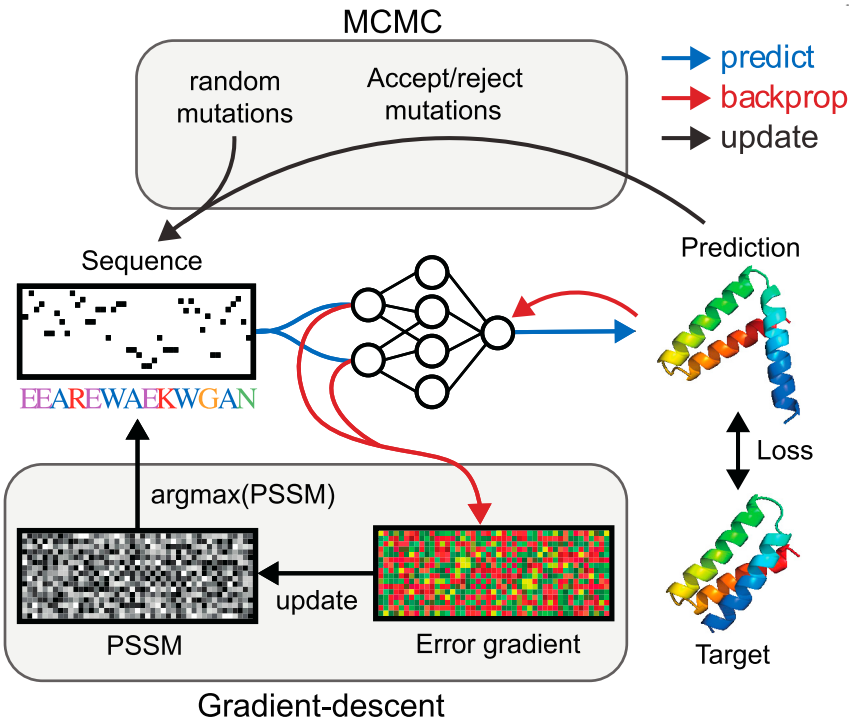

Figure 5. Design pipelines inspired by deep learning structure prediction algorithms

In MCMC-based generation, random mutations are introduced into an initial sequence and the prediction is scored using a loss function. The mutated sequence is accepted when the loss decreases or the Metropolis acceptance criterion is met. These steps are repeated for many iterations resulting in a sequence that satisfies the loss function. Gradient descent-based generation (red arrows) calculates the gradient of the input over the loss, which results in an error gradient matrix. This matrix is used to optimize a PSSM from which the most likely sequence can be extracted for the next iteration. Several rounds of optimization result in convergence of the PSSM from which probable protein sequences can be sampled.

2,000 designs to naturally occurring proteins with experimental validation by nuclear magnetic resonance (NMR) spectroscopy and X-ray crystallography.[64] Goverde et al. targeted a diverse set of protein topologies[74] and created soluble analogs of integral membrane proteins.[68] They were able to show the viability of their computational designs experimentally. Other groups demonstrated that the same approach can successfully create stable scaffolds around discontinuous binding sites from signaling proteins[66] and diverse cyclic homo-oligomers up to 1,550 residues and C33 symmetry when equipped with additional symmetry losses.[76] Wang et al.[6] used a combination of constrained hallucination and missing information recovery or *inpainting* to optimize a scaffold around a functional site and applied their approach to designing *de novo* scaffolds around a respiratory syncytial virus F (RSV-F) epitope, a COVID-19 receptor trap, metal-binding proteins, enzyme active sites, and protein-binding proteins, which were further experimentally validated.

Despite these positive results, low *in vitro* success rates[76] and poor solubility due to surface-exposed hydrophobic residues have been reported,[74,76] requiring downstream refinement of the designs using other sequence design tools such as Rosetta or ProteinMPNN,[26] the latter will be discussed below.

A fundamentally different way to tackle the "inverse folding" problem is to train models that explicitly predict sequences given a structure, thus approximating p(sequence|structure). ProteinMPNN,[26] for example, is a graph-based neural network for sequence design derived from a model developed by Ingraham et al.[25] A backbone encoder module computes node and edge features based on a spatial nearest neighbor graph. A subsequent decoder module updates the node features using the context of previously generated amino acids and converts them into a categorical probability distribution from which a new amino acid type is sampled. To design a new sequence for a given fixed backbone, the model thus predicts one amino acid identity at a time in an autoregressive manner. Since many of the interactions between residues in a 3D protein structure correspond to long-range dependencies in sequence,[77] high performance of ProteinMPNN can be primarily attributed to the graph encoding of the 3D backbone structure, which enables spatial neighborhood aggregation. The authors as well as other researchers who quickly adopted the tool reported high experimental success rates on a variety of protein design tasks, such as monomeric proteins, symmetrically repeated structures, nanoparticles, and targeted protein binders.[26,76,78]

Among other deep learning models for explicit inverse folding, there is a broad consensus on the general setup: they predict a residue type based on some structural representation of its local environment, and they use training datasets which almost exclusively comprise rigid protein structures—often crystal structures. These models therefore differ mainly in architectural choices. Many recent methods are graph based[79–82] and put focus on strong geometric priors.[18,82,83] Some methods predict side chain rotamers in addition to amino acid identities which is not strictly required for sequence design but can be used as auxiliary information about the full atomic structure for assessing and optimizing potential solutions.[84] Despite the considerable amount of attention given to architecture design, some research also highlights the role of data. Hsu et al.[83] employed a data augmentation strategy in which the AlphaFold2[39] structure prediction model was used to create an almost three orders of magnitude larger training set of sequence-structure pairs leading to improved perplexity and sequence recovery.

Apart from ProteinMPNN, most of these tools have not found wide application yet as the validation did not go beyond *in silico* experiments and rather narrow metrics like "sequence recovery." Furthermore, the absence of standardized benchmarks to compare various protein design tools makes it challenging to determine an optimal approach for a specific use case and thereby limits the exploration of the full potential of current methods. We discuss this issue in more detail in the discussion section.

Function to structure

In many protein design applications, the target backbone is not known and must be generated *de novo*. Considering tailored functions, the main goal here is to design novel proteins by optimizing selected structures that can perform specific tasks or exhibit particular properties, such as enzymatic activity, binding affinity to a target molecule, solubility, or stability. Many works to date framed this task as a distribution learning problem and utilized one of the probabilistic frameworks discussed in the "deep generative models" section. As depicted in Figure 6, applications of generative models (VAEs, GANs, and diffusion models) span the whole range of protein design subtasks such as *de novo* backbones,[22,85] small molecule or protein binders,[85] symmetric oligomers,[85] epitope-specific antibodies,[86,87] and motif-supporting scaffolds.[88] Often a strong focus is put on allowing users to steer the generative process and thus design proteins according to their own specifications.

Until recently, structure-centric protein design was mainly addressed with VAEs[86,89] and GANs[21,22] applied to pairwise distance maps, which are a convenient and crucially rotation/translation invariant protein representation amenable to the then predominant CNN architectures. To obtain structural models, these contact maps had to be converted to 3D coordinates using tools as diverse as convex optimization algorithms,[21,90] specialized neural network modules,[22] or other external software tools.[89,91] Oftentimes, modeling of the entire protein structure space with a single neural network proves difficult so that many methods resort to specialized models for specific protein families with limited structural diversity.[86,89] User control is achieved through techniques such as residue masking, latent space interpolation and gradient-guided latent space optimization.[22,86]

A common issue of these early approaches was invalidly generated contact maps, i.e., inconsistent Euclidean distance matrices that cannot be embedded in 3D space. Since the neural networks' output domain is usually not sufficiently restricted, the models have to learn all geometric constraints purely from data, which often leads to imperfect results and ultimately obstructs the reconstruction of 3D models. While this problem can be sidestepped with hybrid models that directly produce 3D coordinates but compute training losses based on 2D pair representations,[86] almost all of the latest structure generating models operate exclusively in 3D. Alongside this trend, researchers have also started to favor the diffusion model framework over VAEs and GANs due to convincing results in the image domain.[92] To the best of our knowledge, there is no fundamental reason why VAEs and GANs have been used primarily with distance and angle maps while diffusion models typically generate structures directly in 3D. We speculate that this phenomenon is an artifact of several coinciding trends in the computational biology community. VAEs and GANs were the predominant probabilistic frameworks when pairwise distance and angle maps have been the protein representation of choice, whereas diffusion models surpassed the other generative models around the same time the community started to shift toward explicit 3D representations and geometric deep learning.

Although diffusion models have been successfully applied to generating small molecules in various settings,[93–98] it is more challenging to apply such techniques to protein design due to the much higher number of atoms in protein structures. To date, diffusion-based methods for protein design overcome this issue by either designing only parts of proteins or by operating on a more coarse-grained protein representation. The most widely adopted choice are C_α atoms together with corresponding residue types[88] as fundamental building blocks. In most cases, this representation is additionally endowed with a notion of global residue orientation based on the $N - C\alpha - C$ backbone atoms, which form a local reference frame.[85,87,99–101] In each of these models, positions, orientations, and amino acid types are equipped with a suitable diffusion process and loss function. Coordinates, for instance, can be noised with Gaussian kernels in $\mathbb{R}^3$, whereas angular features are subject to periodic constraints and thus require bespoke diffusion processes on groups or manifolds.[102,103] Amino acid types are categorical features and as such often combined with a discrete diffusion formulation.[104] Side chains are usually either omitted and only added in a subsequent sequence design step or, in few cases, modeled with a tailored diffusion process for torsional angles.[94,99] Molecular design with diffusion models is an extremely active research area at the moment. Innovations are not only made in model architectures but also in sampling algorithms. Here, diffusion models offer a great amount of flexibility that researchers set out to explore. ProtDiff,[88] for example, introduces a novel sampling scheme based on particle filtering[105] that can be used to generate approximate conditional samples from an unconditionally trained diffusion model, e.g., for sampling scaffolds conditioned on a given functional motif. This is useful in practice as it reduces the burden of creating new, specialized datasets for conditional generation tasks. Chroma[106] trades off sample diversity for quality in a principled way with a low-temperature sampling approach, and employs classifier guidance[92,107] to give users control over protein shape, symmetry or class, and even enable protein design conditioned on textual inputs.

A concrete example in this category is RFdiffusion,[85] which was supported by extensive experimental validations. RFdiffusion is a confluence of many of the ideas discussed above combined with a pre-trained protein structure prediction model (RoseTTAfold). Together with ProteinMPNN it forms a full-fledged toolbox of related methods for various protein design applications, such as stable monomer design, symmetric oligomer design, scaffold design for functional motifs or enzyme active sites, and protein binder design. Similar to other methods, RFdiffusion operates on backbone frames ($N - C\alpha - C$) applying adequate noising processes to the C_α coordinates and orientation matrices, respectively. The authors found that fine-tuning a trained RoseTTAfold model for structure prediction and *self-conditioning*, a technique in which the previous prediction is provided as additional input to the neural network in each denoising step, are beneficial to performance. For binder design and scaffolding applications, separate conditional RFdiffusion models were trained to generate protein structures given a fixed set of residues (interface hotspots, catalytic site, and input motif). Similar to Chroma, a classifier guidance-inspired technique can be used to steer the sampling process toward desirable solutions, which proved to be useful for modeling enzyme substrates implicitly. The paper contains extensive *in silico* benchmarks and, more importantly, experimental evidence for successful protein design with neural diffusion models.

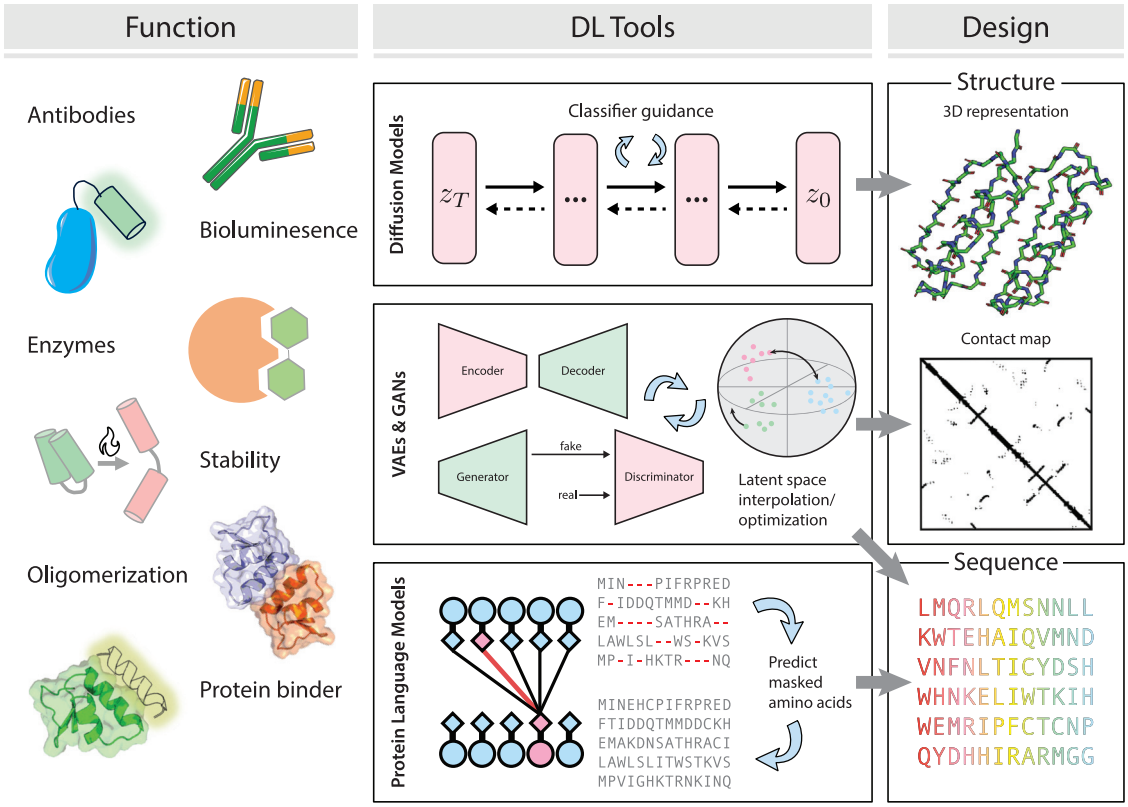

Figure 6. The general workflow of function-to-structure/sequence design

Multiple protein functions can be considered as the starting point including antibody design, protein-based bioluminesence (or luciferases), enzymatic activities, structural stability and folding, designing specific oligomeric states, and protein binders. The popular deep learning methods of choice are VAEs, GANs, diffusion models, and protein language models.

Almost all published diffusion models for protein design limit their evaluations to computational metrics, the most common performance measure being self-consistency (round-trip) evaluation, which assesses whether structures predicted by an *ab initio* folding algorithm are in agreement with the generated backbones after a sequence design step.[22,88,100,101] Unfortunately, the experimental validation of structures is lagging far behind the rapid methodological developments for this family of deep-learning-based protein design tools.

Function to sequence

Unlike structure-centric protein design, function-to-sequence design starts with a desired function or activity and aims to identify the amino acid sequence that can perform that function without explicitly modeling the protein's folded state.

Generative VAE and GAN models have found numerous applications in this domain (Figure 6). Riesselman et al.[108] fitted VAEs to sets of homologous sequences and predicted the effect of mutations by approximating the log-ratio of probabilities of wild-type and mutated sequences. Their model captures non-trivial relationships between protein sequences and can be used to predict beneficial and deleterious mutations that correlate with experimental mutational scans. Hawkins-Hooker et al.[109] used VAEs to approximate the distribution of bacterial luciferase-like proteins at the sequence level and generated new variants by decoding latent vectors from the neighborhood of a target protein. The resulting sequences have been experimentally tested and shown to be functional with up to 35 different positions from any training sequence. Gupta and Zou[110] equipped a GAN model with a potentially non-differentiable scoring module to generate synthetic sequences of small proteins with up to 50 amino acids. By feeding back the most favorable sequences as "real" examples into the training loop at regular intervals, their model is capable of optimizing almost any desired property as long as a reliable scoring oracle exists. Repecka et al.[111] demonstrated with *in vitro* experiments that GANs can generate highly diverse yet functional amino acid sequences of enzymes.

State-of-the-art in protein sequence modeling, however, are large protein language models. A range of works[43,44,112–115] have shown that protein language models are good candidates for efficient exploration of the protein sequence space. For instance, attention maps of transformer-based language models capture the folding structure of the proteins,[114] target protein-binding sites,[114] and are capable of learning residue contacts from the unsupervised language modeling objective[115] without further supervised fine-tuning. These findings recently enabled researchers to employ language models for generating novel protein sequences that satisfy specific functional or structural requirements.

To leverage the semantic organization of the representation space constructed by the language models, a controllable

sequence generation model, ProGen, was introduced for *de novo* protein design.[116] ProGen was trained for the next token prediction task given a list of input protein properties such as biological function or associated family. At inference time, it takes a list of properties as input and iteratively generates protein sequences from scratch. Experimentally validated on five protein families including lysozymes, it has been shown that the artificial proteins expressed as good as the native ones but performed even better in functional assays.

Sgarbossa et al.[117] have shown that protein language models trained on MSAs are highly attractive candidates for sampling novel sequences belonging to specific protein families. In this work, the authors analyzed the MSA Transformer,[118] showed that it is capable of generating sequences that score as well as natural sequences for homology, co-evolution, and structure-based measures, and conclude that the MSA Transformer is a strong candidate for protein sequence generation and protein design. The creators of the MSA Transformer themselves demonstrated in a follow-up work that pre-trained protein language models can capture the functional effects of sequence variation.[119]

Considering a more specific biological target, in a recent publication, Hie et al.[120] reported that protein language models can efficiently evolve human antibodies by suggesting evolutionary plausible mutations. After having performed PLM-based affinity maturation of seven antibodies, the authors managed to improve binding affinity for four clinically relevant antibodies and achieved favorable thermostability and viral neutralization activity against Ebola and SARS-CoV-2.

DISCUSSION

Deep learning has had a transformative effect on the protein design landscape and is arguably the most actively researched family of methods for computational protein design today. While rigorous mathematics often fails to describe the complexity of biological systems and therefore does not lend itself to engineering endeavors, deep learning has proven its effectiveness in finding patterns in complex high dimensional spaces.[121] In this review, we described the latest deep learning techniques available in the toolbox of computational protein designers and how they are currently applied to create (1) sequences for given backbones, (2) new structures for specific purposes, and (3) sequences based on function specifications without explicit modeling of the structure. While there is significant evidence that deep learning approaches will turn out to be the right toolbox for the challenging and yet evolving field of protein design, there are several limitations to consider. In short, these limitations can be categorized into technical difficulties (related to the amount and type of data and the deep learning models), which impose limitations on the biological questions we are currently able to address, and secondly benchmarking challenges.

Data challenges

From a technical point of view, the first challenge is data availability. Deep learning models require a large amount of labeled data for training. However, obtaining such data with decent quality for proteins is challenging and costly. Prominent examples of this problem are protein flexibility and dynamics for which virtually no experimental training data are available, effectively rendering related applications such as protein switches or molecular machines out of scope for current methods. The lack of data beyond static structures even impedes efforts to design proteins in a single rigid state as dynamical properties play a major role in every protein's stability and function. One possible way to mitigate the lack of data limitation are transfer learning techniques where the knowledge or embeddings derived from a pre-trained model can be used to facilitate training of task-specific models where only insufficient amounts of data are available.

Moreover, as outlined in this review, current machine learning systems mainly support rather coarse-grained designs, i.e., at the sequence or backbone structure level. They are so far largely incapable of making accurate predictions at the atomic level, although early-stage solutions for all-atom design have already been put forward.[122] However, higher precision of the structure design tools will be essential to enable explicit side chain conformation modeling without resorting to other tools such as Rosetta or molecular dynamics simulations that come with their own limitations. Without it, success will remain limited for certain targets that require a high level of detail, for instance, small-molecule-binding sites or complementarity-determining regions (CDRs) of antibodies.

Another related question that has not yet been conclusively answered by the field is whether the massive reliance of most machine learning models on protein structures is always a reasonable assumption. Indeed, working at a structural level seemingly provides an inductive bias toward more physically meaningful solutions but at the same time significant parts of the biological reality (solvent interactions, post-translational modifications, dynamics, etc.) are still poorly understood or at least not explicitly modeled in the methods discussed in this review, calling into question the usefulness of this model bias. At our current level of understanding, sequence-centric or hybrid learning approaches could be better suited to certain design tasks thanks to the vast amounts of available clean sequence data. These methods allow us to delegate processing of less well understood components of the biological system to the learning algorithm, which treats them as hidden variables and predicts their phenotypic effects from the sequence data. The work discussed in function to sequence section suggests that many crucial questions regarding functional protein design can already be addressed with purely sequence-based methods that could potentially alleviate the lack of structural data.

Methodological challenges

A broader issue in the field is generalization and out-of-distribution extrapolation of machine learning methods. Since supervised learning algorithms fundamentally fit complicated non-linear functions to the provided data points, their uncertainty increases substantially as we move away from training examples. This has practical implications for designing functional proteins with rare properties. Designing for specific function requires exploring regions of the sequence/structure space that are potentially sparsely supported by natural proteins. Even if deep learning models are trained on the whole natural protein sequence space, they may not be *generalizable* enough to capture such rare properties or functions and fail to design truly *de novo* proteins that are effective and functional. One of the possible remedies

to overcome this obstacle is discussed by Roney and Ovchinnikov[123] where it is shown that AlphaFold2 has learned an approximate biophysical energy function that can be used to predict a protein structure from its primary sequence and thus generalize beyond the available evolutionary data.

As a final technical challenge, we would like to mention *interpretability* of deep learning models. The predominant "black box" approach does not offer any explanations for its predictions and thus hinders targeted interventions by humans. The lack of interpretability also limits the extent to which we gain new insights into the functionality of proteins and prevents us from easily estimating cases where learning-based tools fail before running time-consuming and expensive *in vitro* experiments. Some of these issues are addressed with confidence estimation techniques, which are implemented in an increasing number of data-driven protein design tools, but more work is required to improve the robustness of computational protein design.

Benchmarking challenges

The second major challenge for deep learning methods in protein design is the validation strategy. Researchers utilize different design targets as "test beds" to evaluate the performance of their models. These targets are generally selected based on their relevance to practical use cases or the challenges they pose. Common design targets that deep learning-based approaches can so far address include: structural properties such as stability and folding,[124] binding activity such as designing proteins with high specificity toward targets, or designing inhibitors to disrupt existing protein-protein interactions (e.g., between host and pathogen),[7,78,125] improved enzyme catalytic activity,[111,126–128] and designing antibodies ranging from scaffold design to epitope recognition or affinity maturation toward a specific target.[129,130]

Many of the methods covered in this review have been published as purely computational studies with *in silico* validation. In some of these works, researchers have used AlphaFold2 for the validation of the results. Clearly, *in vitro* evidence of their effectiveness is needed to demonstrate their applicability in practice. While this applies to all categories of models discussed above, the largest gap between methodological improvements and experimental results arguably exists in the generative modeling domain. The field as a whole, and especially diffusion models, are evolving at an incredible pace both in machine learning and computational biology communities but limited experimental evidence of its usefulness in protein design campaigns exists to date. This problem is exacerbated by a lack of rigorous benchmarks that would enable fair comparisons between different methods and protein design strategies. Among the few papers that have experimentally verified their deep-learning-based protein designs, rather vague definitions of experimental success rates (e.g., successful expression, folding or target binding) are often considered as the primary measure of performance. Experimental outcomes are however subject to so many confounding factors that attributing success or failure to the computational design approach alone seems unjustified. We therefore need to develop more convincing and reproducible experiments in the protein design community to gauge and quantify real progress of the field.

Moreover, experimental validation is slow and inaccessible to some research groups. We should therefore also strive for more reliable and comparable *in silico* benchmarks. Current computational analyses are of limited value because neither evaluation metrics nor benchmarking datasets are standardized, which means that new methods cannot be easily compared with existing approaches. Scientific progress can be substantially accelerated if comprehensive and universally accepted benchmarks are established as previously evidenced by the computer vision community at the beginning of the latest AI boom[131] and the critical assessment of structure prediction (CASP) challenge,[132] which led to the development of AlphaFold2.

Outlook

Finally, we would like to briefly speculate on the future of deep learning for protein design. Among the diversity of other machine learning strategies and techniques relevant to protein design that we did not cover in this review, we would like to mention two directions. First, we highlight reinforcement learning (RL)[133] as a powerful paradigm for optimization in highly complex settings. The main idea behind RL is to train an agent to learn making optimal decisions while interacting with a dynamic environment. Through interaction with the environment, the agent can make actions which will receive feedback (reward) from the environment. The goal of RL algorithms is to maximize the expected long-term reward. This class of algorithms has already led to extraordinary breakthroughs in areas as diverse as game playing,[134,135] robotics,[136] plasma control for nuclear fusion,[137] and algorithm discovery,[138] but it is still underexplored as a tool for protein design. While early works on this topic exist to address function-to-sequence design,[139,140] we expect a surge of RL applications as more powerful and efficient molecular simulators become available.

Second, in addition to *de novo* protein design, deep learning and specifically generative models recently tackled challenges in directed evolution-based protein design by advancing the expensive traditional *in vitro* screening *in silico*.[141,142] This is mostly done by building surrogate models of protein fitness landscapes,[143,144] which represent the sequence-to-function relationship. It has a crucial impact on the success rate of the directed evolution approach where proteins usually are designed by searching this space through iterative rounds of mutagenesis. Initially supervised deep learning approaches were used by training over labeled data to understand the sequence-function relationship. As such labeled data are limited and costly, some recent works tried to take advantage of information embedded into unsupervised learning approaches or even combine both.[145] With the recent outstanding performance of language models, they can now encode the astronomical sequence-function space using unlabeled data. The main outcome here is to have access to millions of protein sequences from growing databases without requiring careful function assignment. Such models can then be considered as pre-trained components to provide embeddings for downstream supervised tasks. In our opinion, deep learning for directed evolution has a high potential to become much more efficient, yet as effective as the classical experimental methods for designing new functional protein sequences, specifically with rare functions.

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.cels.2023.10.006.

ACKNOWLEDGMENTS

H.K. was supported by the French Agence Nationale de la Recherche (ANR), under grant ANR-22-CPJ2-0075-01. I.I. has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Marie Sklodowska-Curie grant agreement No. 945363. A.S. was supported by Microsoft Research AI4Science. B.C. was supported by the Swiss National Science Foundation, the NCCR in Chemical Biology, the NCCR in Molecular Systems Engineering, and the ERC Starting grant No. 716058.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

1. Huang, P.S., Boyken, S.E., and Baker, D. (2016). The coming of age of de novo protein design. Nature *537*, 320–327. https://doi.org/10.1038/nature19946.

2. Siegel, J.B., Zanghellini, A., Lovick, H.M., Kiss, G., Lambert, A.R., St Clair, J.L., Gallaher, J.L., Hilvert, D., Gelb, M.H., Stoddard, B.L., et al. (2010). Computational design of an enzyme catalyst for a stereoselective bimolecular Diels-Alder reaction. Science *329*, 309–313. https://doi.org/10.1126/science.1190239.

3. Nanda, V., and Koder, R.L. (2010). Designing artificial enzymes by intuition and computation. Nat. Chem. *2*, 15–24. https://doi.org/10.1038/nchem.473.

4. Tinberg, C.E., Khare, S.D., Dou, J., Doyle, L., Nelson, J.W., Schena, A., Jankowski, W., Kalodimos, C.G., Johnsson, K., Stoddard, B.L., et al. (2013). Computational design of ligand-binding proteins with high affinity and selectivity. Nature *501*, 212–216. https://doi.org/10.1038/nature12443.

5. Jha, R.K., Leaver-Fay, A., Yin, S., Wu, Y., Butterfoss, G.L., Szyperski, T., Dokholyan, N.V., and Kuhlman, B. (2010). Computational design of a pak1 binding protein. J. Mol. Biol. *400*, 257–270. https://doi.org/10.1016/j.jmb.2010.05.006.

6. Wang, J., Lisanza, S., Juergens, D., Tischer, D., Watson, J.L., Castro, K.M., Ragotte, R., Saragovi, A., Milles, L.F., Baek, M., et al. (2022). Scaffolding protein functional sites using deep learning. Science *377*, 387–394. https://doi.org/10.1126/science.abn2100.

7. Gainza, P., Wehrle, S., Van Hall-Beauvais, A., Marchand, A., Scheck, A., Harteveld, Z., Buckley, S., Ni, D., Tan, S., Sverrisson, F., et al. (2023). De novo design of protein interactions with learned surface fingerprints. Nature *617*, 176–184. https://doi.org/10.1038/s41586-023-05993-x.

8. Fleishman, S.J., Whitehead, T.A., Ekiert, D.C., Dreyfus, C., Corn, J.E., Strauch, E.M., Wilson, I.A., and Baker, D. (2011). Computational design of proteins targeting the conserved stem region of influenza hemagglutinin. Science *332*, 816–821. https://doi.org/10.1126/science.1202617.

9. Castro, K.M., Scheck, A., Xiao, S., and Correia, B.E. (2022). Computational design of vaccine immunogens. Curr. Opin. Biotechnol. *78*, 102821. https://doi.org/10.1016/j.copbio.2022.102821.

10. Sesterhenn, F., Yang, C., Bonet, J., Cramer, J.T., Wen, X., Wang, Y., Chiang, C.I., Abriata, L.A., Kucharska, I., Castoro, G., et al. (2020). De novo protein design enables the precise induction of RSV-neutralizing antibodies. Science *368*, eaay5051. https://doi.org/10.1126/science.aay5051.

11. Arnold, F.H. (1998). Design by directed evolution. Acc. Chem. Res. *31*, 125–131. https://doi.org/10.1021/ar960017f.

12. Yue, K., and Dill, K.A. (1992). Inverse protein folding problem: designing polymer sequences. Proc. Natl. Acad. Sci. USA *89*, 4163–4167. https://doi.org/10.1073/pnas.89.9.4163.

13. He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. 31st Conference on Neural Information Processing Systems. Advances in Neural Information Processing Systems 30 (NIPS 2017).

15. Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: pre-training of deep bidirectional transformers for language understanding. https://doi.org/10.48550/arXiv.1810.04805.

16. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. Adv. Neural Inf. Process. Syst. *33*, 1877–1901.

17. LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. Nature *521*, 436–444. https://doi.org/10.1038/nature14539.

18. Jing, B., Eismann, S., Suriana, P., Townshend, R.J., and Dror, R. (2020). Learning from protein structure with geometric vector perceptrons. https://doi.org/10.48550/arXiv.2009.01411.

19. Wang, X., Terashi, G., Christoffer, C.W., Zhu, M., and Kihara, D. (2020). Protein docking model evaluation by 3d deep convolutional neural networks. Bioinformatics *36*, 2113–2118. https://doi.org/10.1093/bioinformatics/btz870.

20. Li, Z., Nguyen, S.P., Xu, D., and Shang, Y. (2017). Protein loop modeling using deep generative adversarial network. 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI) pp, 1085–1091. https://doi.org/10.1109/ICTAI.2017.00166.

21. Anand, N., and Huang, P. (2018). Generative modeling for protein structures. Adv. Neural Inf. Process. Syst. *31*, 7504–7515.

22. Anand, N., Eguchi, R., and Huang, P. (2019). Fully differentiable full-atom protein backbone generation. ICLR 2019 Workshop DeepGenStruct.

23. Yang, J., Anishchenko, I., Park, H., Peng, Z., Ovchinnikov, S., and Baker, D. (2020). Improved protein structure prediction using predicted interresidue orientations. Proc. Natl. Acad. Sci. USA *117*, 1496–1503. https://doi.org/10.1073/pnas.1914677117.

24. Bronstein, M.M., Bruna, J., Cohen, T., and Veličković, P. (2021). Geometric deep learning: grids, groups, graphs, geodesics, and gauges. https://doi.org/10.48550/arXiv.2104.13478.

25. Ingraham, J., Garg, V., Barzilay, R., and Jaakkola, T. (2019). Generative models for graph-based protein design. Conference on Neural Information Processing Systems.

26. Dauparas, J., Anishchenko, I., Bennett, N., Bai, H., Ragotte, R.J., Milles, L.F., Wicky, B.I.M., Courbet, A., de Haas, R.J., Bethel, N., et al. (2022). Robust deep learning–based protein sequence design using proteinmpnn. Science *378*, 49–56. https://doi.org/10.1126/science.add2187.

27. Gainza, P., Sverrisson, F., Monti, F., Rodolà, E., Boscaini, D., Bronstein, M.M., and Correia, B.E. (2020). Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. Nat. Methods *17*, 184–192. https://doi.org/10.1038/s41592-019-0666-6.

28. Sverrisson, F., Feydy, J., Correia, B.E., and Bronstein, M.M. (2021). Fast end-to-end learning on protein surfaces. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15267–15276. https://doi.org/10.1109/CVPR46437.2021.01502.

29. Somnath, V.R., Bunne, C., and Krause, A. (2021). Multi-scale representation learning on proteins. Adv. Neural Inf. Process. Syst. *34*, 25244–25255.

30. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., and Dahl, G.E. (2017). Neural message passing for quantum chemistry. In International Conference on Machine Learning, pp. 1263–1272.

31. Kipf, T.N., and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. https://doi.org/10.48550/arXiv.1609.02907.

32. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2017). Graph attention networks. https://doi.org/10.48550/arXiv.1710.10903.

33. Mitchell, T.M. (1980). The need for biases in learning generalizations. http://dml.cs.byu.edu/~cgc/docs/mldm_tools/Reading/Need%20for%20Bias.pdf.

34. Batzner, S., Musaelian, A., Sun, L., Geiger, M., Mailoa, J.P., Kornbluth, M., Molinari, N., Smidt, T.E., and Kozinsky, B. (2022). E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. Nat. Commun. *13*, 2453. https://doi.org/10.1038/s41467-022-29939-5.

35. Thomas, N., Smidt, T., Kearnes, S., Yang, L., Li, L., Kohlhoff, K., and Riley, P. (2018). Tensor field networks: rotation-and translation-equivariant neural networks for 3d point clouds. https://doi.org/10.48550/arXiv.1802.08219.

36. Kondor, R. (2018). N-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. https://doi.org/10.48550/arXiv.1803.01588.

37. Anderson, B., Hy, T.S., and Kondor, R. (2019). Cormorant: covariant molecular neural networks. https://doi.org/10.48550/arXiv.1906.04015.

38. Schütt, K.T., Sauceda, H.E., Kindermans, P.-J., Tkatchenko, A., and Müller, K.-R. (2018). Schnet - a deep learning architecture for molecules and materials. J. Chem. Phys. *148*, 241722. https://doi.org/10.1063/1.5019779.

39. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with alphafold. Nature *596*, 583–589. https://doi.org/10.1038/s41586-021-03819-2.

40. UniProt Consortium (2019). UniProt: a worldwide hub of protein knowledge. Nucleic Acids Res. *47*, D506–D515. https://doi.org/10.1093/nar/gky1049.

41. Heinzinger, M., Elnaggar, A., Wang, Y., Dallago, C., Nechaev, D., Matthes, F., and Rost, B. (2019). Modeling aspects of the language of life through transfer-learning protein sequences. BMC Bioinformatics *20*, 723. https://doi.org/10.1186/s12859-019-3220-8.

42. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog *1*, 9.

43. Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C.L., Ma, J., et al. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proc. Natl. Acad. Sci. USA *118*, e2016239118. https://doi.org/10.1073/pnas.2016239118.

44. Elnaggar, A., Heinzinger, M., Dallago, C., Rihawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., et al. (2020). Prottrans: towards cracking the language of life's code through self-supervised deep learning and high performance computing. https://doi.org/10.48550/arXiv.2007.06225.

45. Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J., Abbeel, P., and Song, Y.S. (2019). Evaluating protein transfer learning with tape. Adv. Neural Inf. Process. Syst. *32*, 9689–9701. https://doi.org/10.48550/arXiv.1906.08230.

46. Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. Neural Comput. *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

47. Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: encoder-decoder approaches. https://doi.org/10.48550/arXiv.1409.1259.

48. Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. https://doi.org/10.48550/arXiv.1412.3555.

49. Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y.N. (2017). Convolutional sequence to sequence learning. In International Conference on Machine Learning, pp. 1243–1252. https://doi.org/10.48550/arXiv.1705.03122.

50. Ba, J.L., Kiros, J.R., and Hinton, G.E. (2016). Layer normalization. https://doi.org/10.48550/arXiv.1607.06450.

51. Bishop, C.M., and Nasrabadi, N.M. (2006). Pattern Recognition and Machine Learning (Springer).

52. Kingma, D.P., and Welling, M. (2013). Auto-encoding variational Bayes. https://doi.org/10.48550/arXiv.1312.6114.

53. Hinton, G.E., and Salakhutdinov, R.R. (2006). Reducing the dimensionality of data with neural networks. Science *313*, 504–507. https://doi.org/10.1126/science.1127647.

54. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks. https://doi.org/10.48550/arXiv.1406.2661.

55. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2020). Generative adversarial networks. Commun. ACM *63*, 139–144. https://doi.org/10.1145/3422622.

56. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In Proceedings of the 32nd International Conference on Machine Learning, PMLR, *37*, pp. 2256–2265. https://doi.org/10.48550/arXiv.1503.03585.

57. Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. Adv. Neural Inf. Process. Syst. *33*, 6840–6851. https://doi.org/10.48550/arXiv.2006.11239.

58. Leaver-Fay, A., Tyka, M., Lewis, S.M., Lange, O.F., Thompson, J., Jacak, R., Kaufman, K.W., Renfrew, P.D., Smith, C.A., Sheffler, W., et al. (2011). Rosetta3: an object-oriented software suite for the simulation and design of macromolecules. Methods Enzymol. *487*, 545–574. https://doi.org/10.1016/B978-0-12-381270-4.00019-6.

59. Simons, K.T., Kooperberg, C., Huang, E., and Baker, D. (1997). Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions. J. Mol. Biol. *268*, 209–225. https://doi.org/10.1006/jmbi.1997.0959.

60. Kuhlman, B., and Baker, D. (2000). Native protein sequences are close to optimal for their structures. Proc. Natl. Acad. Sci. USA *97*, 10383–10388. https://doi.org/10.1073/pnas.97.19.10383.

61. Kuhlman, B., Dantas, G., Ireton, G.C., Varani, G., Stoddard, B.L., and Baker, D. (2003). Design of a novel globular protein fold with atomic-level accuracy. Science *302*, 1364–1368. https://doi.org/10.1126/science.1089427.

62. Leman, J.K., Weitzner, B.D., Lewis, S.M., Adolf-Bryfogle, J., Alam, N., Alford, R.F., Aprahamian, M., Baker, D., Barlow, K.A., Barth, P., et al. (2020). Macromolecular modeling and design in rosetta: recent methods and frameworks. Nat. Methods *17*, 665–680. https://doi.org/10.1038/s41592-020-0848-2.

63. Senior, A.W., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., Qin, C., Žídek, A., Nelson, A.W.R., Bridgland, A., et al. (2020). Improved protein structure prediction using potentials from deep learning. Nature *577*, 706–710. https://doi.org/10.1038/s41586-019-1923-7.

64. Anishchenko, I., Pellock, S.J., Chidyausiku, T.M., Ramelot, T.A., Ovchinnikov, S., Hao, J., Bafna, K., Norn, C., Kang, A., Bera, A.K., et al. (2021). De novo protein design by deep network hallucination. Nature *600*, 547–552. https://doi.org/10.1038/s41586-021-04184-w.

65. Norn, C., Wicky, B.I.M., Juergens, D., Liu, S., Kim, D., Tischer, D., Koepnick, B., Anishchenko, I.; Foldit Players, and Baker, D., et al. (2021). Protein sequence design by conformational landscape optimization. Proc. Natl. Acad. Sci. USA *118*, e2017228118. https://doi.org/10.1073/pnas.2017228118.

66. Tischer, D., Lisanza, S., Wang, J., Dong, R., Anishchenko, I., Milles, L.F., Ovchinnikov, S., and Baker, D. (2020). Design of proteins presenting discontinuous functional sites using deep learning. https://doi.org/10.1101/2020.11.29.402743.

67. Frank, C.J., Khoshouei, A., de Stigter, Y., Schiewitz, D., Feng, S., Ovchinnikov, S., and Dietz, H. (2023). Efficient and scalable de novo protein design using a relaxed sequence space. https://doi.org/10.1101/2023.02.24.529906.

68. Goverde, C.A., Pacesa, M., Dornfeld, L.J., Georgeon, S., Rosset, S., Dauparas, J., Shellhaas, C., Kozlov, S., Baker, D., Ovchinnikov, S., et al. (2023). Computational design of soluble analogues of integral membrane protein structures. https://doi.org/10.1101/2023.05.09.540044.

69. Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G.R., Wang, J., Cong, Q., Kinch, L.N., Schaeffer, R.D., et al. (2021). Accurate prediction of protein structures and interactions using a three-track neural network. Science *373*, 871–876. https://doi.org/10.1126/science.abj8754.

70. Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. (2023). Evolutionary-scale prediction of atomic-level protein structure with a language model. Science *379*, 1123–1130. https://doi.org/10.1126/science.ade2574.

71. Wu, R., Ding, F., Wang, R., Shen, R., Zhang, X., Luo, S., Su, C., Wu, Z., Xie, Q., Berger, B., et al. (2022). High-resolution de novo structure prediction from primary sequence. https://doi.org/10.1101/2022.07.21.500999.

72. Verkuil, R., Kabeli, O., Du, Y., Wicky, B.I., Milles, L.F., Dauparas, J., Baker, D., Ovchinnikov, S., Sercu, T., and Rives, A. (2022). Language models generalize beyond natural proteins. https://doi.org/10.1101/2022.12.21.521521.

73. Hie, B., Candido, S., Lin, Z., Kabeli, O., Rao, R., Smetanin, N., Sercu, T., and Rives, A. (2022). A high-level programming language for generative protein design. https://doi.org/10.1101/2022.12.21.521526.

74. Goverde, C.A., Wolf, B., Khakzad, H., Rosset, S., and Correia, B.E. (2023). De novo protein design by inversion of the alphafold structure prediction network. Protein Sci. *32*, e4653. https://doi.org/10.1002/pro.4653.

75. Zhang, Y., and Skolnick, J. (2005). Tm-align: A protein structure alignment algorithm based on the tm-score. Nucleic Acids Res. *33*, 2302–2309. https://doi.org/10.1093/nar/gki524.

76. Wicky, B.I.M., Milles, L.F., Courbet, A., Ragotte, R.J., Dauparas, J., Kinfu, E., Tipps, S., Kibler, R.D., Baek, M., DiMaio, F., et al. (2022). Hallucinating symmetric protein assemblies. Science *378*, 56–61. https://doi.org/10.1126/science.add1964.

77. Marks, D.S., Colwell, L.J., Sheridan, R., Hopf, T.A., Pagnani, A., Zecchina, R., and Sander, C. (2011). Protein 3d structure computed from evolutionary sequence variation. PLoS One *6*, e28766. https://doi.org/10.1371/journal.pone.0028766.

78. Bennett, N.R., Coventry, B., Goreshnik, I., Huang, B., Allen, A., Vafeados, D., Peng, Y.P., Dauparas, J., Baek, M., Stewart, L., et al. (2023). Improving de novo protein binder design with deep learning. Nat. Commun. *14*, 2625. https://doi.org/10.1038/s41467-023-38328-5.

79. Gao, Z., Tan, C., and Li, S.Z. (2022). Alphadesign: a graph protein design method and benchmark on Alphafolddb. https://doi.org/10.48550/arXiv.2202.01079.

80. Gao, Z., Tan, C., and Li, S.Z. (2023). Pifold: toward effective and efficient protein inverse folding. International Conference on Learning Representations. https://doi.org/10.48550/arXiv.2209.12643.

81. Zhou, X., Chen, G., Ye, J., Wang, E., Zhang, J., Mao, C., Li, Z., Hao, J., Huang, X., Tang, J., et al. (2023). Protein sequence design by entropy-based iterative refinement. https://doi.org/10.1101/2023.02.04.527099.

82. Mao, W., Zhu, M., Chen, H., and Shen, C. (2023). Modeling protein structure using geometric vector field networks. https://doi.org/10.1101/2023.05.07.539736.

83. Hsu, C., Verkuil, R., Liu, J., Lin, Z., Hie, B., Sercu, T., Lerer, A., and Rives, A. (2022). Learning inverse folding from millions of predicted structures. In International Conference on Machine Learning, pp. 8946–8970.

84. Anand, N., Eguchi, R., Mathews, I.I., Perez, C.P., Derry, A., Altman, R.B., and Huang, P.-S. (2022). Protein sequence design with a learned potential. Nat. Commun. *13*, 746. https://doi.org/10.1038/s41467-022-28313-9.

85. Watson, J.L., Juergens, D., Bennett, N.R., Trippe, B.L., Yim, J., Eisenach, H.E., Ahern, W., Borst, A.J., Ragotte, R.J., Milles, L.F., et al. (2023). De novo design of protein structure and function with RFdiffusion. Nature *620*, 1089–1100. https://doi.org/10.1038/s41586-023-06415-8.

86. Eguchi, R.R., Choe, C.A., and Huang, P.-S. (2022). Ig-vae: generative modeling of protein structure by direct 3d coordinate generation. PLoS Comp. Biol. *18*, e1010271. https://doi.org/10.1371/journal.pcbi.1010271.

87. Luo, S., Su, Y., Peng, X., Wang, S., Peng, J., and Ma, J. (2022). Antigen-specific antibody design and optimization with diffusion-based generative models. Adv. Neural Inf. Process. Syst. *35*, 9754–9767.

88. Trippe, B.L., Yim, J., Tischer, D., Broderick, T., Baker, D., Barzilay, R., and Jaakkola, T. (2022). Diffusion probabilistic modeling of protein backbones in 3d for the Motif-Scaffolding problem. https://doi.org/10.48550/arXiv.2206.04119.

89. Guo, X., Du, Y., Tadepalli, S., Zhao, L., and Shehu, A. (2020). Generating tertiary protein structures via an interpretative variational autoencoder. https://doi.org/10.48550/arXiv.2004.07119.

90. Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. *3*, 1–122. https://doi.org/10.1561/2200000016.

91. Adhikari, B., Bhattacharya, D., Cao, R., and Cheng, J. (2015). Confold: residue-residue contact-guided ab initio protein folding. Proteins *83*, 1436–1449. https://doi.org/10.1002/prot.24829.

92. Dhariwal, P., and Nichol, A. (2021). Diffusion models beat gans on image synthesis. Adv. Neural Inf. Process. Syst. *34*, 8780–8794. https://doi.org/10.48550/arXiv.2105.05233.

93. Shi, C., Luo, S., Xu, M., and Tang, J. (2021). Learning gradient fields for molecular conformation generation. International Conference on Machine Learning, 9558–9568. https://doi.org/10.48550/arXiv.2105.03902.

94. Jing, B., Corso, G., Chang, J., Barzilay, R., and Jaakkola, T. (2022). Torsional diffusion for molecular conformer generation. https://doi.org/10.48550/arXiv.2206.01729.

95. Hoogeboom, E., Satorras, V.G., Vignac, C., and Welling, M. (2022). Equivariant Diffusion for Molecule Generation. In 3rd International Conference on Machine Learning, 8867–8887. https://doi.org/10.48550/arXiv.2203.17003.

96. Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S., and Tang, J. (2022). Geodiff: a geometric diffusion model for molecular conformation generation. https://doi.org/10.48550/arXiv.2203.02923.

97. Igashov, I., Stärk, H., Vignac, C., Satorras, V.G., Frossard, P., Welling, M., Bronstein, M., and Correia, B. (2022). Equivariant 3D-conditional diffusion models for molecular linker design. https://doi.org/10.48550/arXiv.2210.05274.

98. Schneuing, A., Du, Y., Harris, C., Jamasb, A., Igashov, I., Du, W., Blundell, T., Lió, P., Gomes, C., Welling, M., et al. (2022). Structure-based drug design with equivariant diffusion models. https://doi.org/10.48550/arXiv.2210.13695.

99. Anand, N., and Achim, T. (2022). Protein structure and sequence generation with equivariant denoising diffusion probabilistic models. https://doi.org/10.48550/arXiv.2205.15019.

100. Lin, Y., and AlQuraishi, M. (2023). Generating novel, designable, and diverse protein structures by equivariantly diffusing oriented residue clouds. https://doi.org/10.48550/arXiv.2301.12485.

101. Yim, J., Trippe, B.L., De Bortoli, V., Mathieu, E., Doucet, A., Barzilay, R., and Jaakkola, T. (2023). Se(3) diffusion model with application to protein backbone generation. https://doi.org/10.48550/arXiv.2302.02277.

102. De Bortoli, V., Mathieu, E., Hutchinson, M., Thornton, J., Teh, Y.W., and Doucet, A. (2022). Riemannian score-based generative modeling. https://doi.org/10.48550/arXiv.2202.02763.

103. Leach, A., Schmon, S.M., Degiacomi, M.T., and Willcocks, C.G. (2022). Denoising diffusion probabilistic models on so(3) for rotational alignment. In ICLR 2022 Workshop on Geometrical and Topological Representation Learning.

104. Austin, J., Johnson, D.D., Ho, J., Tarlow, D., and van den Berg, R. (2021). Structured denoising diffusion models in discrete state-spaces. Adv. Neural Inf. Process. Syst. *34*, 17981–17993. https://doi.org/10.48550/arXiv.2107.03006.

105. Doucet, A., De Freitas, N., Gordon, N.J., et al. (2001). Sequential Monte Carlo Methods in Practice (Springer).

106. Ingraham, J., Baranov, M., Costello, Z., Frappier, V., Ismail, A., Tie, S., Wang, W., Xue, V., Obermeyer, F., Beam, A., et al. (2022). Illuminating protein space with a programmable generative model. https://doi.org/10.1101/2022.12.01.518682.

107. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., and Poole, B. (2020). Score-based generative modeling through stochastic differential equations. https://doi.org/10.48550/arXiv.2011.13456.

108. Riesselman, A.J., Ingraham, J.B., and Marks, D.S. (2018). Deep generative models of genetic variation capture the effects of mutations. Nat. Methods *15*, 816–822. https://doi.org/10.1038/s41592-018-0138-4.

109. Hawkins-Hooker, A., Depardieu, F., Baur, S., Couairon, G., Chen, A., and Bikard, D. (2021). Generating functional protein variants with variational autoencoders. PLoS Comp. Biol. *17*, e1008736. https://doi.org/10.1371/journal.pcbi.1008736.

110. Gupta, A., and Zou, J. (2019). Feedback gan for dna optimizes protein functions. Nat. Mach. Intell. *1*, 105–111. https://doi.org/10.1038/s42256-019-0017-4.

111. Repecka, D., Jauniskis, V., Karpus, L., Rembeza, E., Rokaitis, I., Zrimec, J., Poviloniene, S., Laurynenas, A., Viknander, S., Abuajwa, W., et al. (2021). Expanding functional protein sequence spaces using generative adversarial networks. Nat. Mach. Intell. *3*, 324–333. https://doi.org/10.1038/s42256-021-00310-5.

112. Bepler, T., and Berger, B. (2019). Learning protein sequence embeddings using information from structure. https://doi.org/10.48550/arXiv.1902.08661.

113. Bepler, T., and Berger, B. (2021). Learning the protein language: evolution, structure, and function. Cell Syst. *12*, 654–669.e3. https://doi.org/10.1016/j.cels.2021.05.017.

114. Vig, J., Madani, A., Varshney, L.R., Xiong, C., Socher, R., and Rajani, N.F. (2020). Bertology meets biology: interpreting attention in protein language models. https://doi.org/10.48550/arXiv.2006.15222.

115. Rao, R., Meier, J., Sercu, T., Ovchinnikov, S., and Rives, A. (2020). Transformer protein language models are unsupervised structure learners. https://doi.org/10.1101/2020.12.15.422761.

116. Madani, A., Krause, B., Greene, E.R., Subramanian, S., Mohr, B.P., Holton, J.M., Olmos, J.L., Jr., Xiong, C., Sun, Z.Z., Socher, R., et al. (2023). Large language models generate functional protein sequences across diverse families. Nat. Biotechnol. *41*, 1099–1106. https://doi.org/10.1038/s41587-022-01618-2.

117. Sgarbossa, D., Lupo, U., and Bitbol, A.-F. (2023). Generative power of a protein language model trained on multiple sequence alignments. eLife *12*, e79854. https://doi.org/10.7554/eLife.79854.

118. Rao, R.M., Liu, J., Verkuil, R., Meier, J., Canny, J., Abbeel, P., Sercu, T., and Rives, A. (2021). Msa transformer. In International Conference on Machine Learning, pp. 8844–8856. https://doi.org/10.1101/2021.02.12.430858.

119. Meier, J., Rao, R., Verkuil, R., Liu, J., Sercu, T., and Rives, A. (2021). Language models enable zero-shot prediction of the effects of mutations on protein function. Adv. Neural Inf. Process. Syst. *34*, 29287–29303. https://doi.org/10.1101/2021.07.09.450648.

120. Hie, B.L., Shanker, V.R., Xu, D., Bruun, T.U.J., Weidenbacher, P.A., Tang, S., Wu, W., Pak, J.E., and Kim, P.S. (2023). Efficient evolution of human antibodies from general protein language models. Nat. Biotechnol. https://doi.org/10.1038/s41587-023-01763-2.

121. AlQuraishi, M., and Sorger, P.K. (2021). Differentiable biology: using deep learning for biophysics-based and data-driven modeling of molecular mechanisms. Nat. Methods *18*, 1169–1180. https://doi.org/10.1038/s41592-021-01283-4.

122. Chu, A.E., Cheng, L., El Nesr, G., Xu, M., and Huang, P.-S. (2023). An all-atom protein generative model. https://doi.org/10.1101/2023.05.24.542194.

123. Roney, J.P., and Ovchinnikov, S. (2022). State-of-the-art estimation of protein model accuracy using alphafold. Phys. Rev. Lett. *129*, 238101. https://doi.org/10.1103/PhysRevLett.129.238101.

124. Singer, J.M., Novotney, S., Strickland, D., Haddox, H.K., Leiby, N., Rocklin, G.J., Chow, C.M., Roy, A., Bera, A.K., Motta, F.C., et al. (2022). Large-scale design and refinement of stable proteins using sequence-only models. PLoS One *17*, e0265020. https://doi.org/10.1371/journal.pone.0265020.

125. Cao, L., Coventry, B., Goreshnik, I., Huang, B., Sheffler, W., Park, J.S., Jude, K.M., Markovic, I., Kadam, R.U., Verschueren, K.H., et al. (2022). Design of protein-binding proteins from the target structure alone. Nature *605*, 551–560. https://doi.org/10.1038/s41586-022-04654-9.

126. Ming, Y., Wang, W., Yin, R., Zeng, M., Tang, L., Tang, S., and Li, M. (2023). A review of enzyme design in catalytic stability by artificial intelligence. Brief. Bioinform. *24*, bbad065. https://doi.org/10.1093/bib/bbad065.

127. Li, F., Yuan, L., Lu, H., Li, G., Chen, Y., Engqvist, M.K.M., Kerkhoven, E.J., and Nielsen, J. (2022). Deep learning-based k cat prediction enables improved enzyme-constrained model reconstruction. Nat. Cat. *5*, 662–672. https://doi.org/10.1038/s41929-022-00798-z.

128. Yeh, A.H.-W., Norn, C., Kipnis, Y., Tischer, D., Pellock, S.J., Evans, D., Ma, P., Lee, G.R., Zhang, J.Z., Anishchenko, I., et al. (2023). De novo design of luciferases using deep learning. Nature *614*, 774–780. https://doi.org/10.1038/s41586-023-05696-3.

129. Hummer, A.M., Abanades, B., and Deane, C.M. (2022). Advances in computational structure-based antibody design. Curr. Opin. Struct. Biol. *74*, 102379. https://doi.org/10.1016/j.sbi.2022.102379.

130. Dai, B., and Bailey-Kellogg, C. (2021). Protein interaction interface region prediction by geometric deep learning. Bioinformatics *37*, 2580–2588. https://doi.org/10.1093/bioinformatics/btab154.

131. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In IEEE conference on computer vision and pattern recognition, pp. 248–255. https://doi.org/10.1109/CVPR.2009.5206848.

132. Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K., and Moult, J. (2021). Critical assessment of methods of protein structure prediction (casp)—round xiv. Proteins *89*, 1607–1617. https://doi.org/10.1002/prot.26237.

133. Sutton, R.S., and Barto, A.G. (2018). Reinforcement Learning: an Introduction (MIT Press).

134. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. Science *362*, 1140–1144. https://doi.org/10.1126/science.aar6404.

135. Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Choi, D.H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. Nature *575*, 350–354. https://doi.org/10.1038/s41586-019-1724-z.

136. Andrychowicz, O.M., Baker, B., Chociej, M., Józefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., et al. (2020). Learning dexterous in-hand manipulation. Int. J. Robot. Res. *39*, 3–20. https://doi.org/10.1177/0278364919887447.

137. Degrave, J., Felici, F., Buchli, J., Neunert, M., Tracey, B., Carpanese, F., Ewalds, T., Hafner, R., Abdolmaleki, A., de Las Casas, D., et al. (2022). Magnetic control of Tokamak plasmas through deep reinforcement learning. Nature *602*, 414–419. https://doi.org/10.1038/s41586-021-04301-9.

138. Fawzi, A., Balog, M., Huang, A., Hubert, T., Romera-Paredes, B., Barekatain, M., Novikov, A., R Ruiz, F.J.R., Schrittwieser, J., Swirszcz, G., et al. (2022). Discovering faster matrix multiplication algorithms with reinforcement learning. Nature *610*, 47–53. https://doi.org/10.1038/s41586-022-05172-4.

139. Angermueller, C., Dohan, D., Belanger, D., Deshpande, R., Murphy, K., and Colwell, L. (2019). Model-based reinforcement learning for biological sequence design. In International conference on learning representations.

140. Feng, L., Nouri, P., Muni, A., Bengio, Y., and Bacon, P.-L. (2022). Designing biological sequences via meta-reinforcement learning and bayesian optimization. https://doi.org/10.48550/arXiv.2209.06259.

141. Yang, K.K., Wu, Z., and Arnold, F.H. (2019). Machine-learning-guided directed evolution for protein engineering. Nat. Methods *16*, 687–694. https://doi.org/10.1038/s41592-019-0496-6.

142. Wittmann, B.J., Johnston, K.E., Wu, Z., and Arnold, F.H. (2021). Advances in machine learning for directed evolution. Curr. Opin. Struct. Biol. *69*, 11–18. https://doi.org/10.1016/j.sbi.2021.01.008.

143. Notin, P., Dias, M., Frazer, J., Hurtado, J.M., Gomez, A.N., Marks, D., and Gal, Y. (2022). Tranception: protein fitness prediction with autoregressive transformers and inference-time retrieval. In International Conference on Machine Learning, pp. 16990–17017. https://doi.org/10.48550/arXiv.2205.13760.

144. Wright, S. (1932). The Roles of Mutation, Inbreeding, Crossbreeding, and Selection in Evolution (Blackwell Publishing).

145. Hsu, C., Nisonoff, H., Fannjiang, C., and Listgarten, J. (2022). Learning protein fitness models from evolutionary and assay-labeled data. Nat. Biotechnol. *40*, 1114–1122. https://doi.org/10.1038/s41587-021-01146-5.