Protein Engineering, Design and Selection, 2024, **37**, gzad024
https://doi.org/10.1093/protein/gzad024
Advance access publication date 29 December 2023
Review

Protein sequence design on given backbones with deep learning

Yufeng Liu [1] **and Haiyan Liu** [1,2,3,*]

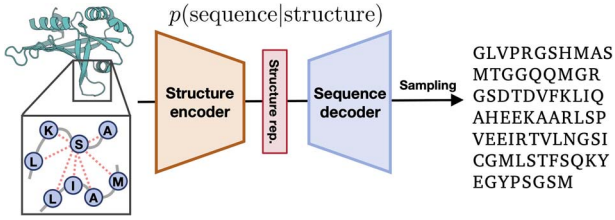
[1]MOE Key Laboratory for Membraneless Organelles and Cellular Dynamics, School of Life Sciences, Division of Life Sciences and Medicine, University of Science and Technology of China, Hefei, Anhui 230027, China
[2]Biomedical Sciences and Health Laboratory of Anhui Province, University of Science and Technology of China, Hefei, Anhui 230027, China
[3]School of Biomedical Engineering, Suzhou Institute for Advanced Research, University of Science and Technology of China, Suzhou, Jiangsu 215004, China

*To whom correspondence should be addressed. E-mail: e-mails:hyliu@ustc.edu.cn

Abstract

Deep learning methods for protein sequence design focus on modeling and sampling the many- dimensional distribution of amino acid sequences conditioned on the backbone structure. To produce physically foldable sequences, inter-residue couplings need to be considered properly. These couplings are treated explicitly in iterative methods or autoregressive methods. Non-autoregressive models treating these couplings implicitly are computationally more efficient, but still await tests by wet experiment. Currently, sequence design methods are evaluated mainly using native sequence recovery rate and native sequence perplexity. These metrics can be complemented by sequence-structure compatibility metrics obtained from energy calculation or structure prediction. However, existing computational metrics have important limitations that may render the generalization of computational test results to performance in real applications unwarranted. Validation of design methods by wet experiments should be encouraged.

Graphical Abstract



Keywords: computational protein design, de novo protein design, inverse folding

Introduction

Computational protein design (CPD) aims at creating protein of tailored structure and function to fulfill needs in wide areas from therapeutics (Reig *et al.*, 2012; Huang *et al.*, 2016) to bio-catalysis (Siegel *et al.*, 2010; Cui *et al.*, 2021). One fundamental problem of CPD is fixed-backbone sequence design or inverse protein folding (Dahiyat and Mayo, 1997; Leaver-Fay *et al.*, 2011; Xiong *et al.*, 2014; O'Connell *et al.*, 2018), which is to find amino acid sequences that physically fold into a target peptide backbone. In protein engineering, methods for fixed-backbone sequence design can be applied to perform structure-based redesign of natural proteins (Cui *et al.*, 2021; Liu *et al.*, 2021; Lu *et al.*, 2022). These methods can also be applied to select amino acid sequences for *de novo* backbones (Kuhlman *et al.*, 2003; Polizzi and DeGrado, 2020).

In the last three decades, a range of methods for fixed-backbone sequence design (simply referred to as sequence design below) have been proposed, from the earlier conventional methods based on energy functions (Dahiyat and Mayo, 1997; Kuhlman *et al.*, 2003; Leaver-Fay *et al.*, 2011; Simonson *et al.*, 2013; Xiong *et al.*, 2014; Mackenzie *et al.*, 2016; Huang *et al.*, 2020; Liang *et al.*, 2022) and the more recent approaches using deep learning (Li *et al.*, 2014; O'Connell *et al.*, 2018; Ingraham *et al.*, 2019; Qi and Zhang, 2020; Anand *et al.*, 2022; Dauparas *et al.*, 2022; Li *et al.*, 2022; Liu *et al.*, 2022; Gao *et al.*, 2022a; Huang *et al.*, 2023; McPartlon and Xu, 2023; Yan *et al.*, 2023; Zhou *et al.*, 2023).

In conventional sequence design methods, amino acid sidechain types are selected by minimizing empirical energy functions that depend on both the sidechain type and the sidechain conformation (Dahiyat and Mayo, 1997; Kuhlman *et al.*, 2003). The sidechain conformations are usually represented as discrete rotamers (Shapovalov and Dunbrack, 2011) while the energy functions are usually linear combinations of relatively simple one-body and two-body terms, which can be either physics-based (*e.g.* Rosetta Design (Leaver-Fay *et al.*, 2011), Proteus (Simonson *et al.*, 2013))or

Received: August 16, 2023. **Revised:** December 8, 2023. **Accepted:** December 18, 2023

statistically learned from data (*e.g.* ABACUS (Xiong *et al.*, 2014; Xiong *et al.*, 2020), TERM (Mackenzie *et al.*, 2016)) (we note that a method based on parameter fitting for a series expansion scoring function was recently reported (Liang *et al.*, 2022)). Although several function-based sequence design methods have been verified by experimentally solved protein structures (Leaver-Fay *et al.*, 2011; Xiong *et al.*, 2014; Xiong *et al.*, 2020), even the best-performing models still exhibit relatively low success rates (Schreier *et al.*, 2009; Fleishman *et al.*, 2011) and high sensitivity to the target structures (Johansson *et al.*, 2016; Marin *et al.*, 2021) despite significant research efforts.

Deep learning methods for sequence design (Li *et al.*, 2014; O'Connell *et al.*, 2018; Ingraham *et al.*, 2019; Qi and Zhang, 2020; Anand *et al.*, 2022; Dauparas *et al.*, 2022; Li *et al.*, 2022; Liu *et al.*, 2022; Gao *et al.*, 2022a; Huang *et al.*, 2023; McPartlon and Xu, 2023; Yan *et al.*, 2023; Zhou *et al.*, 2023) do not rely on explicit energy functions. Instead, they focused on modeling and sampling the many-dimensional sequence distribution conditioned on the backbone structure. Compared with conventional energy function-based approaches, several deep learning methods have been demonstrated to perform superiorly over conventional energy function-based methods not only in computational tests, but also in wet experiments (Dauparas *et al.*, 2022; Liu *et al.*, 2022).

The focus of this review will be on deep learning sequence design methods. Energy function-based methods will only be briefly overviewed. In examining the various methods, we will pay special attention to how they represent the physically important inter-residue couplings. As a large number of recent sequence design methods have been evaluated only in computational benchmarks, we will also discuss the advantages and limitations of current computational metrics.

Sequence design by minimizing energy functions

By using discrete rotamers (Shapovalov and Dunbrack, 2011) to represent the sidechain conformations, Mayo and coworkers have first shown that the problem of structure-based sequence design could be solved through combinatorial optimization of an empirical energy function in the joint sidechain type and rotamer space (Dahiyat and Mayo, 1997). Clearly, the energy function plays a central role in this approach. Energy functions for sequence design are usually composed of physics-based terms and statistically learned terms. Some methods used mainly physics-based energy terms (for example, RosettaDesign (Leaver-Fay *et al.*, 2011), Proteus (Simonson *et al.*, 2013) and EvoEF (Huang *et al.*, 2020)), while other methods rely more on statistically learned terms (for example, ABACUS (Xiong *et al.*, 2014), TERM (Mackenzie *et al.*, 2016)). Based on the physics-based Rosetta energy function (Leaver-Fay *et al.*, 2011), Baker and coworkers have developed the most well-known RosttaDesign method, which has been utilized to achieve a series of milestone works in *de novo* protein design (Huang *et al.*, 2016; Kuhlman and Bradley, 2019). Experimentally solved structures of proteins designed by statistically learned energy functions have also been reported (Xiong *et al.*, 2014; Mackenzie *et al.*, 2016; Huang *et al.*, 2022). These studies demonstrated that besides physics-based approaches, general data-driven models could also be developed to solve the *de novo* sequence design problem.

Despite significant progress, energy function-based sequence design has suffered from several important limitations. One is that the discrete rotamers can only represent the continuous side chain conformations approximately. The other is that the complicated, many-body molecular interactions contributing to the total free energy of a folded protein are only represented as a linear combination of one-body and two-body terms in simplified forms (Leaver-Fay *et al.*, 2011; Simonson *et al.*, 2013; Xiong *et al.*, 2014; Mackenzie *et al.*, 2016). These limitations caused conventional energy function-based *de novo* sequence design to exhibit unsatisfactorily low success rates (Xiong *et al.*, 2014; Johansson *et al.*, 2016; Marin *et al.*, 2021) in practical applications.

Sequence design by maximizing probabilities

Sequence design can be described as finding a mapping from the peptide backbone structure space to the amino acid sequence space. We note the backbone structure as X, the overall sequence as S, and the sidechain type at position i in S as S_i. Treating the sidechain types at different positions as random variables, a natural way of defining the structure-to-sequence mapping is to maximize the multi-dimensional conditional probability $P(S|X)$. The number of dimensions of $P(S|X)$, being equal to the length of the peptide chain (noted as L below), is usually large, making it challenging to model $P(S|X)$ directly. To enable effective modeling, $P(S|X)$can be marginalized or factorized into lower dimensional probability distributions.

The simplest lower dimensional distributions are the set of 1-D position-specific sidechain type preferences, which collectively form a sequence profile. The sidechain type preference for position i can be formalized as $P(S|X) = \sum_{j\neq i,\text{and}j\in[1,L]} P(s_1,\ldots,s_i,\ldots,s_L|X)$. In this formulation, the sidechain type at position i is affected by the side chain types at all the other positions only in an averaged way. If sequence selection is based on the maximization of these 1-D marginal probabilities, the advantage is that the sidechain types at individual positions can be selected independently. However, it would be unlikely for the resulting sequence as a whole to be foldable, because the couplings between the sidechain types at different positions have been completely ignored. In fact, all experimentally validated foldable *de novo* protein sequences reported so far (Leaver-Fay *et al.*, 2011; Xiong *et al.*, 2014; Mackenzie *et al.*, 2016; Anand *et al.*, 2022; Dauparas *et al.*, 2022; Liu *et al.*, 2022) have been designed with methods that considered inter-residue coupling. Moreover, the correlated substitutions at structural contacting positions have been extensively analyzed in the sequence-structure relationship of natural proteins (Göbel *et al.*, 1994).

To include the coupling between positions in the 1-D distributions, we can start from the fully conditional probability $P(s_i|s_j \text{ with } j\neq i \text{ and } j\in[1,L],X)$. In this probability, the sidechain type at position i is considered to be dependent on the sidechain types at all the other positions. Based on the structure, we know that the sidechain at position i is in direct contact with not all residues but only a small number of neighboring residues. Thus, we may retain only the neighboring residues in the condition, and consider replacing $P(s_i|s_j \text{ with } j\neq i \text{ and } j\in[1,L],X)$ with $P(s_i|s_j \text{ with } j\in N_i,X)$, in which N_i refers to the set of structural neighbors of residue i. To select amino acid sequences by maximizing $P(s_i|s_j \text{ with } j\in N_i,X)$ is physically

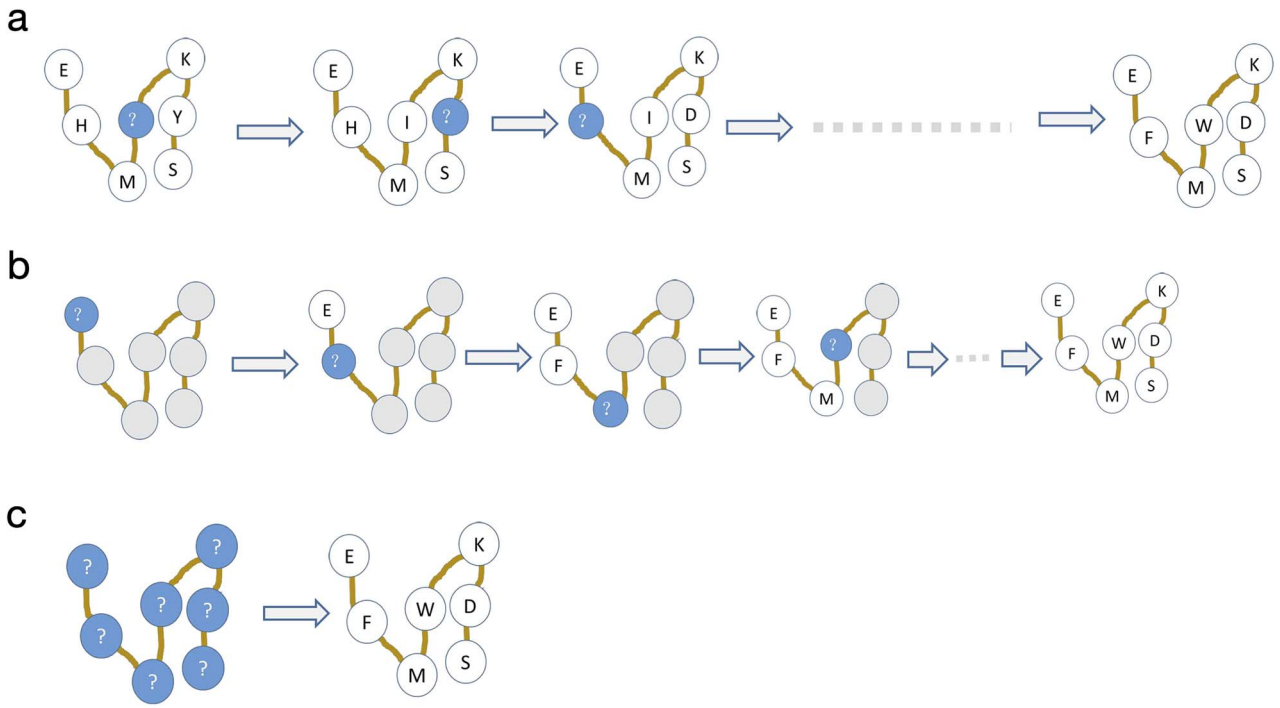


Fig. 1. Different approaches of generating sequences according to conditional probabilities. (**a**) Iterative approach. (**b**) Autoregressive approach. (**c**) Non-autoregressive and non-iterative approach. The circles containing explicit sidechain types indicate positions that constitute the sequence context for the current iteration or autoregression step. The circles containing question mark indicate the positions for which the sidechain types are to be derived from the current context

sound. However, the set of $P\left(s_i|s_j \text{ with } j \in N_i, X\right)$ for all position $i \in [1, L]$ need to be jointly maximized. This can be achieved with an iterative approach (Liu *et al.*, 2022; Huang *et al.*, 2023) (Fig. 1a).

Another way of factorizing the many dimensional $P\left(S|X\right)$ is to use the auto-regressive expansion, namely, $P\left(S|X\right) = \prod_{i=1}^{L} P\left(s_i|s_j \text{ with } j < i \text{ and } j \in [1, L], X\right)$. The individual factors in this expansion are maximized in primary or random sequential order (Fig. 1b) (Ingraham *et al.*, 2019; Dauparas *et al.*, 2022; Hsu *et al.*, 2022). This can save the cost of iterative updating. The lack of information about the sidechain types of environmental residues in the earlier phase of the autoregression process may increase the chance of choosing unsuitable sidechain types (Huszár, 2015; Ranzato *et al.*, 2015). Such errors may accumulate and cannot be corrected without iterative refinements.

In general, the factorization or marginalization of $P\left(S|X\right)$ usually leads to 1-D probabilities of the general form $P\left(\mathrm{s_i}|\mathrm{Context_i}\right)$, in which $\mathrm{Context_i}$ represents the environment or context data that conditionally determines the preferences of sidechain types at position i. Several representative AI-based techniques were listed with their availabilities in Table I.

Representing the context with deep neural networks

In deep learning sequence design, data describing $\mathrm{Context_i}$ are transformed using multiple layers of neural network into a latent representation, which is then projected with more layers of neural network to produce the context-conditioned sidechain type preferences (Fig. 1) (O'Connell *et al.*, 2018; Ingraham *et al.*, 2019; Jing *et al.*, 2020; Anand *et al.*, 2022; Dauparas *et al.*, 2022; Liu *et al.*, 2022; Gao *et al.*, 2022a). According to this schematic view, the overall network can be divided into an encoder part, which is responsible for generating the representation of $\mathrm{Context_i}$, and a decoder part, which is responsible for projecting the representation into sidechain types (Ingraham *et al.*, 2019; Dauparas *et al.*, 2022; Liu *et al.*, 2022).

A main data component of $\mathrm{Context_i}$ is the 3-D backbone structure X. The inherent organization of the 3-D structural data can be encoded using 3-D convolutional neural networks (3D-CNN) (Ji *et al.*, 2012; Qi and Zhang, 2020; Anand *et al.*, 2022) or graph neural networks (GNN) (Kipf and Welling, 2016; Jing *et al.*, 2020; Gao *et al.*, 2022a). The organization of nodes in these networks can be defined to reflect the spatially neighboring relationships between atoms or residues. The $\mathrm{Context_i}$ may also include sequence data, which can be directly attached to the structural data as input to the CNN or GNN (Qi and Zhang, 2020; Anand *et al.*, 2022; Dauparas *et al.*, 2022; Hsu *et al.*, 2022; Liu *et al.*, 2022). Alternatively, the structural and sequence context can be encoded jointly by a 2-D pair representation coupled with a 1-D single representation (Ren *et al.*, 2023), as in AlphaFold2 (Jumper *et al.*, 2021).

The output of a sequence design network should be invariant to any translation or rotation of the backbone structure X (Jing *et al.*, 2020; Hsu *et al.*, 2022). This so-called SE(3)-invariant requirement can be strictly fulfilled either by using SE(3)-equivariant models like the geometric vector perceptron neural network (GVPNN) (Jing *et al.*, 2020) or by featurizing X into SE(3)-invariant quantities such as distances and angles at the input end (Yang *et al.*, 2020). Alternatively, networks that are effectively SE(3)-invariant can be trained with data augmentation, in which randomly translated and rotated structures are repetitively used in training (Anand *et al.*, 2022).

Table I. Different deep leaning based protein sequence design methods

Methods	Sequence design approach	Availability
Ingraham *et al.*	Autoregressive	https://github.com/jingraham/neurips19-graph-protein-design
ESMIF	Autoregressive	https://github.com/facebookresearch/esm
ProteinMPNN	Autoregressive	https://github.com/dauparas/ProteinMPNN
ABACUS-R	Iterative	https://doi.org/10.24433/CO.3351944.v1
ProDESIGN-LE	Iterative	https://github.com/bigict/ProDESIGN-LE
AttnPacker	Iterative	https://github.com/MattMcPartlon/AttnPacker
ProRefiner	Iteratively refine autoregressively generated sequences	https://github.com/veghen/ProRefiner
3DCNN	Optimizing negative pseudo-log-likelihood *via* simulated annealing	https://github.com/ProteinDesignLab/protein_seq_des
PiFold	Non-autoregressive and non-iterative approach	https://github.com/A4Bio/PiFold
CarbonDesign	Optimization of a learned Markovian random field followed by recycling	NA
GRADE-IF	Denoising diffusion probabilistic model	https://github.com/ykiiiiii/GraDe_IF

A sample of deep learning sequence design models

Some of the pioneer methods (Li *et al.*, 2014; Torng and Altman, 2017; O'Connell *et al.*, 2018; Wang *et al.*, 2018; Chen *et al.*, 2019; Qi and Zhang, 2020; Zhang *et al.*, 2020) used deep learning with CNN to predict the sequence profile. These methods were evaluated mainly based on the native sequence recovery rate, or the fraction of positions at which the native residue types were of the largest probabilities.

Examples of methods performing full sequence design by maximizing the set of $P\left(s_i|s_j \text{ with } j \in N_i, X\right)$include 3DCNN, ABACUS-R, and ProDESIGN-LE. These methods encode only the local structural environment of residue i instead of the entire global structure X as the actual structural context. This reduced the computational costs for modeling the individual $P\left(s_i|s_j \text{ with } j \in N_i, X\right)$, allowing the use of iterative updating to design complete sequences.

The ProteinMPNN method (Dauparas *et al.*, 2022) and its prototype model (Ingraham *et al.*, 2019) use the autoregressive factorization of $P(S|X)$ to generate sequences. They use a graph neural network (GNN) with repetitive message passing to encode the entire structure X associated with partial sequence context. In ProteinMPNN, the autoregression is performed in a random instead of a sequential order of residue positions. This may to some extent alleviate the error accumulation problem associated with the autoregressive generation of long sequences. Another model that uses autoregressive factorization is ESM-IF (Hsu *et al.*, 2022), which uses GVPNN to encode the structure X and has been trained with not only experimentally solved protein structures but also AlphaFold2-predicted ones.

Several methods use networks of specially designed architectures to encode the global backbone structure and produce amino acid sequences non-autoregressively (Gao *et al.*, 2022a; Gao *et al.*, 2022b; McPartlon and Xu, 2023) (Fig. 1c. Skipping the autoregressive process in sequence generation can significantly reduce computational costs. However, this also leads to the decoupling of the sidechain types at different positions at the output end of the network. Then, all important inter-position coupling of sidechain types must be implicitly encoded by the networks. In other words, the complete, unfactorized $P(S|X)$ needs to be modeled as a whole. This seems to be a rather challenging task. Moreover, common computational metrics such as the native sequence recovery rate do not measure whether inter-residue couplings have been correctly modeled. Thus, the actual foldability of full sequences designed with these non-autoregressive approaches still awaits experimental tests.

More recently, several models have been proposed to improve deep learning sequence design in several interesting directions. One is to mend the potential problem of error accumulation in autoregression with iterative refinements (Ren *et al.*, 2023; Zhou *et al.*, 2023). The computational cost of further refinements can be effectively reduced by fixing the sidechain types at low entropy positions (Zhou *et al.*, 2023) and training a specialized refinement module. In the recently proposed CarbonDesign method (Ren *et al.*, 2023), the structural context is transformed into a Markovian random field that specifies the single position sidechain type preferences as well as the pairwise inter-position coupling with explicit sidechain type dependencies. The equivalence between the Markovian random field formulation and an effective empirical energy function with one-body and two-body interactions can lend physical interpretability to the resulting model. Additionally, the model proposed in CarbonDesign was trained with pair amino acid prediction loss, which has been proved with its improvement for the performance of generated sequences (both in sequence recovery rate and amino acid distribution) by an ablation study.

Currently, the diversities of high confidence sequences designed by most deep learning methods are still quite restricted in comparison with the large variability observed in natural protein sequences. A graph denoising diffusion model for inverse folding has been proposed to generate more diverse sequences of comparable confidence (Yi *et al.*, 2023).

Another emerging direction is to combine structure-based inverse folding with sequence-based protein language models (Zheng *et al.*, 2023). Protein language models including ESM1b (Rives *et al.*, 2021) and ESM2 (Lin *et al.*, 2022) can complement structure-based models by encoding extensive evolution information related to function conservation (Ferruz and Höcker, 2022). They can also provide sequence information for structurally non-deterministic regions.

Metrics for evaluating sequence design methods

While computational metrics of designed sequences have been comprehensively reviewed previously (Li *et al.*, 2013), most studies have considered the native sequence recovery rate

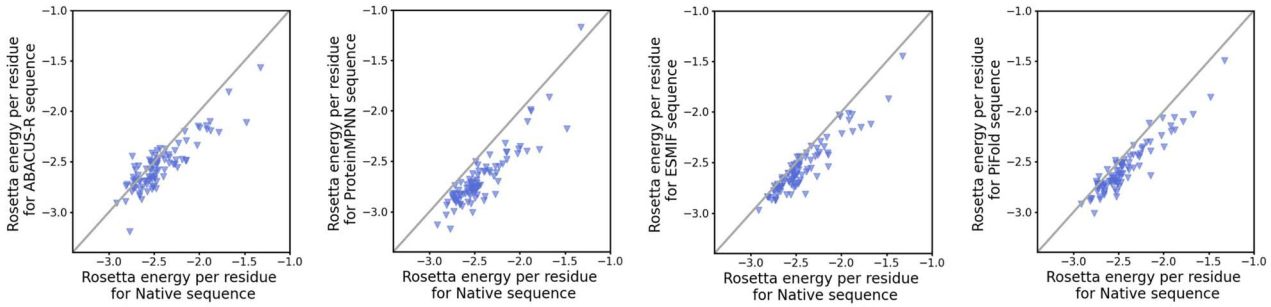


Fig. 2. Scattering plots of per-residue Rosetta energies computed for sequences designed by various methods *versus* those computed for the native sequences. The energies in the Rosetta unit have been calculated on the relaxed structures. From left to right, the sequence design methods considered are ABACUS-R, ProteinMPNN, ESM-IF and PiFold

(meaning the fraction of positions at which the designed sidechain type agreed with the corresponding native type) as the main metric for evaluating the performance of sequence design models (Ingraham *et al.*, 2019; Jing *et al.*, 2020; Hsu *et al.*, 2022; Gao *et al.*, 2022a). This is undoubtedly a very useful metric. However, it has several important limitations. First, because it is the main objective for network training, this metric is prone to distortions caused by over training or information leaking from the training data to test data. Considering the huge amount of data that needs to be processed and the complicated evolutionary relationships between protein sequences, the partitioning of the training and test data sets would not be perfect, which could cause information leaking. Biophysically, protein sequences of less than 30% sequence identity can already fold into very similar backbone structures. On the other hand, the native sequence recovery rate reported for various deep learning methods ranged from slightly over 30% (Ingraham *et al.*, 2019) to close to 60% (Gao *et al.*, 2022a) depending on the benchmark set. Thus, simply pursuing an even higher native sequence recovery rate may not be an important goal for structure-based sequence design.

Another limitation of the metric of native sequence recovery rate is that it considers the designed sidechain types at individual positions separately. No inter-position coupling is considered. As inter-residue interactions are key to the stability of folded protein structures, the native sequence recovery rate alone cannot measure the actual foldability of designed sequences.

A related metric of the native sequence recovery rate is the perplexity of native sequences (Ingraham *et al.*, 2019; Jing *et al.*, 2020; Hsu *et al.*, 2022). It measures the model predicted probabilities of native sidechain types at individual positions. Compared with the native sequence recovery rate, this metric evaluates the predicted marginal probability distributions at a more fine-grained level. However, as the native sequence recovery rate, this metric is also prone to over training and the inter-residue couplings are still not evaluated.

Sequence design methods can also be evaluated by examining the compatibility of the designed sequences with the target structures with existing computational methods. One way of doing this is to consider the Rosetta energies of the designed protein, which can be compared between different methods as well as with those of the natural proteins (Leaver-Fay *et al.*, 2011; Anand *et al.*, 2022; Liu *et al.*, 2022). Another way is to carry out structure prediction (with AlphaFold2 (Jumper *et al.*, 2021), for example) on the designed sequences and compare the predicted structures with the corresponding target structure. Such structural similarity is usually measured by the self-consistency template modeling-score (scTM-score) (Zhang and Skolnick, 2004) or self-consistency root mean square deviation (scRMSD) value (Dauparas *et al.*, 2022; Liu *et al.*, 2022; Ren *et al.*, 2023). The predicted local Distance Difference Test (plDDT) scores (Jumper *et al.*, 2021) of the predictions may also to some extent represent the foldability of the designed sequences.

A major limitation of the energy metric or the structure prediction is that current computational tools are not really sensitive to small errors in the designed sequences. The per-residue Rosetta energies are averages over a large number of residues, each contributing a sum of a large number of terms. In the final per-residue values, a few unfavorable local interactions caused by small design errors can easily be overshadowed by noises. Similarly, the AlphaFold2 predictions are also known to be insensitive to point mutations that can disrupt protein folding.

In Figs 2–4, we compared the above sequence-structure compatibility metrics for several sequence design methods on a random set of 90 natural backbones that covered different fold classes (see Table II for their PDB IDs). We emphasize that these results are only meant for illustrative purposes and should by no means be treated as systematic benchmarks. The tested methods include ABACUS-R (Liu *et al.*, 2022) (an iterative method), ProteinMPNN (Dauparas *et al.*, 2022) (an autoregressive method), ESM-IF (Hsu *et al.*, 2022) (an autoregressive method trained with both experimentally determined and computationally predicted protein structures), and PiFold (Gao *et al.*, 2022a) (a non-autoregressive method).

Results in figs 2–4 suggest that although there may be some differences among ABACUS-R, ProteinMPNN and ESM-IF, no statistically significant conclusion can be drawn about the relative performances of two different methods in most cases given the limited amount of data: except that the sequences designed by ProteinMPNN and PiFold shows statistically significant difference in their distributions of pLDDT (P-value: 0.006) and their distributions of the scRMSD (P-value: 0.010), the remaining P-values are from 0.085 to 0.999 for the distributions of pLDDT generated by two different models to be distinguishable and from 0.176 to 0.799 for the distributions of scRMSD to be distinguishable. Moreover, we note that ABACUS-R and ProteinMPNN are among the few methods that are already backed by experimentally-solved protein structures. Interestingly, ESM-IF and PiFold do not exhibit discernible superiority over the other two models by the sequence-structure compatibility metrics despite their higher native sequence recovery rates.

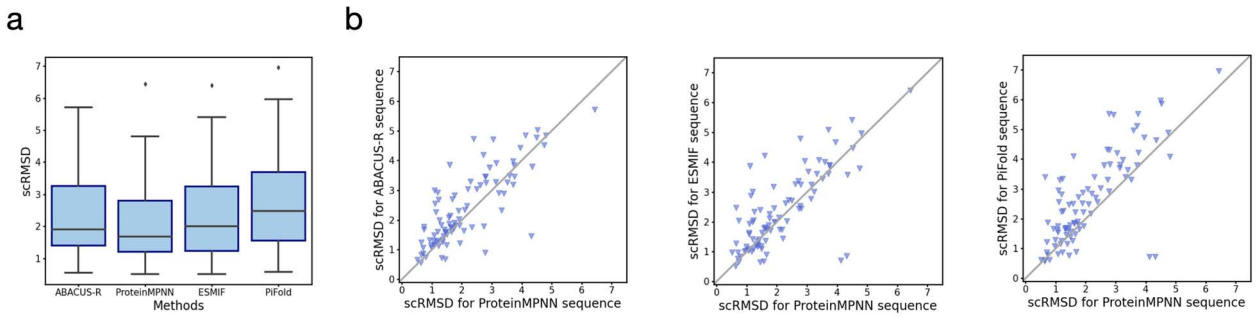


Fig. 3. Similarity between the AlphaFold2 predicted structures for sequences designed by different methods and the target backbones. The designs are performed on 90 target backbones. (**a**) Comparison of the scRMSDs of different design methods. (**b**) Scattering plots of the scRMSDs of different methods *versus* ProteinMPNN. From left to right ABACUS-R, ESM-IF and PiFold

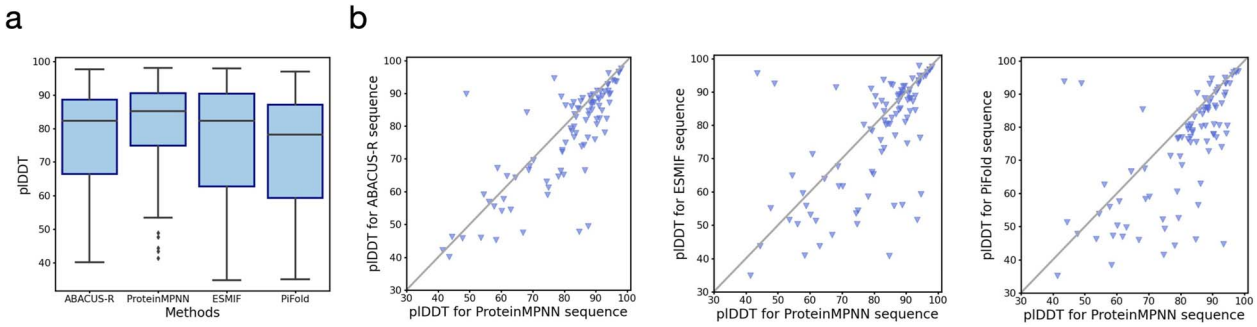


Fig. 4. The pIDDT scores of AlphaFold2 predicted structures for sequences designed by different methods on 90 target backbones. (**a**) Comparison of the pIDDT scores of different design methods. (**b**) Scattering plots of the pIDDT scores of different methods *versus* ProteinMPNN. From left to right ABACUS-R, ESM-IF and PiFold

Table II. The PDB IDs and chain IDs of 90 natural backbones used for testing different methods

Fold classes[a]	PDB ID and chain IDs of backbones[b]
all-a	2rfm_A, 3nrx_A, 4phj_A, 1dm5_A, 4bvx_B, 5f9e_A, 4g09_A, 4p09_A, 2fyu_K, 2g3b_A, 4kjm_A, 6eou_A, 3h20_A, 5a1n_B, 1se0_A, 5d8c_B, 1rx0_A, 6trj_A, 5h1x_A, 6rnk_A, 5nm4_A, 4oe9_A, 5ghe_A, 6fdd_A
all-β	3vyp_A, 3nap_A, 3zn6_A, 4j27_A, 4f8l_A, 4ywa_A, 5vxz_C, 4hh8_A, 2j8k_A, 2d37_A, 2h2b_A, 6qdi_A, 5dyq_A, 5e6x_A
aβ	4q05_A, 5a0y_B, 3g0s_A, 3g7u_A, 5v01_A, 6iq1_A, 1r6w_A, 4z9x_A, 2hhj_A, 2iya_A, 1fxd_A, 2ixs_A, 3tos_A, 3mq2_A, 1zsy_A, 6t7o_A, 4f3q_A, 2eo4_A, 1vyr_A, 4b5o_A, 4kvf_A, 6iik_A, 5hmb_A, 2wb0_X, 3na8_A, 3hsk_A, 3t7k_A, 3e8p_A, 3nvs_A, 1twu_A, 2v3z_A, 3oti_A, 2f6i_A, 3nyt_A, 2wm8_A, 3i99_A, 3oo8_A, 2f5x_A, 3esh_A, 5nhm_A, 3nk6_A, 3zv4_A, 3czc_A, 5svy_A, 4yn3_B, 6m09_B, 5mk9_A, 4whn_A, 5d8m_A, 4xyw_A, 4xe7_A

[a]The fold classes for the natural backbones were assigned by CATH (Sillitoe *et al.*, 2021). [b]These natural backbones were randomly selected from the test set of model$_{eval}$ in (Liu *et al.*, 2022). The sequence identities among each other are less than 50%.

Summary

To perform sequence design with deep learning, the many-dimensional probability of the amino acid sequence conditioned on the backbone structure can be factorized into 1-D conditional distributions. The important inter-residue coupling is treated in iterative or autoregressive manners. In non-iterative and non-autoregressive models, inter-residue coupling must be represented implicitly by the network, which is fundamentally challenging.

Existing computational metrics for evaluating sequence design methods are useful, but with limitations. The most widely used native sequence recovery rate metric is prone to over training. It also cannot tell whether inter-residue couplings have been included properly. Sequence-structure compatibility can be computationally evaluated through energy calculation or structure prediction. However, current methods for this purpose are still insensitive to small errors in designed sequences.

The ultimate purpose of computational protein design is to create real protein molecules with desired properties. The most concerning trait of a sequence design method is perhaps its success rate in real-world applications. In contrast to the rapidly increasing number of studies reporting models improving on the computational metrics, only a few studies have reported results of wet experiments testing computational designs. More solid progress in computational protein design calls for more and closer collaborations between dry and wet experiments.

Details on running and evaluating different methods

To fairly compare the performance of different methods, independent runs were performed for each natural backbone using different methods. We provide the details on running and evaluating as follows.

ProteinMPNN: We used the ProteinMPNN (default) model with 0.2Å noise. We used the testing scripts of ProteinMPNN from the ProteinMPNN GitHub repository (https://github.com/dauparas/ProteinMPNN). all parameter settings employ the default options provided by GitHub. For each natural backbone, three sequences were independently designed with default random seeds.

ABACUS-R: We used the test script provided on the CodeOcean (https://doi.org/10.24433/CO.3351944.v1) for protein sequence design. All parameters were selected from the default options provided in the config file on the website. For each natural backbone, three sequences were independently designed with default random seeds.

ESM-IF: We used the test script provided in the ESM GitHub repository (https://github.com/facebookresearch/esm/tree/main/examples/inverse_folding), with the parameter file named esm_if1_gvp4_t16_142M_UR50 and all other default settings. For each natural backbone, three sequences were independently designed.

PiFold: We used the source code and the test scripts provided in PiFold GitHub repository (https://github.com/A4Bio/PiFold). The parameters were downloaded from another repository (https://zenodo.org/records/8031783). All parameter settings employed the default options provided in the source code. Because PiFold software independently predicts the distribution of the amino acid for each residue, thus the amino acid types with the highest probability for each residue were selected to form a designed amino acid sequence per backbone for further analysis.

The Rosetta energy of designed sequences and native sequences were computed using the RosettaDesign program v.3.12 (Leaver-Fay *et al*., 2011). As all of these methods do not explicitly reconstruct sidechain conformations, all sidechains of all of the evaluated sequences (that is, the native sequences and the designed sequences) have been repacked by using the FixBB option of Rosetta, and then the complete structures have been relaxed using Rosetta RelaxBB to determine the minimized energies.

We predict the protein structures with AlphaFold2 provided in the AlphaFold2 GitHub repository (https://github.com/deepmind/alphafold). When predicting the structure of designed sequences, we only searched for templates from PDB for use by AlphaFold2, while multiple sequence alignment was not used. The predicted backbones and natural backbones were superimposed with TMalign (https://zhanggroup.org/TM-score), The average values of Rosetta energy, scRMSD and pLDDT for each natural backbone were analyzed and compared.

Author contributions

All authors performed a literature search, HY Liu and YF Liu designed the experimental study. YF Liu performed the calculations under the supervision of HY Liu, all authors wrote and edited the paper.

Conflict of interest

None declared.

Funding

This work was supported by the National Natural Science Foundation of China (22177107, T2221005, 92253302).

References

Anand, N., Eguchi, R., Mathews, I.I. *et al.* (2022) *Nat. Commun.*, **13**, 746. https://doi.org/10.1038/s41467-022-28313-9.

Chen, S., Sun, Z., Lin, L. *et al.* (2019) *J. Chem. Inf. Model.*, **60**, 391–399. https://doi.org/10.1021/acs.jcim.9b00438.

Cui, Y., Chen, Y., Liu, X. *et al.* (2021) *ACS Catal.*, **11**, 1340–1350. https://doi.org/10.1021/acscatal.0c05126.

Dahiyat, B.I. and Mayo, S.L. (1997) *Science*, **278**, 82–87. https://doi.org/10.1126/science.278.5335.82.

Dauparas, J., Anishchenko, I., Bennett, N. *et al.* (2022) *Science*, **378**, 49–56. https://doi.org/10.1126/science.add2187.

Ferruz, N. and Höcker, B. (2022) *Nat. Mach. Intell.*, **4**, 521–532. https://doi.org/10.1038/s42256-022-00499-z.

Fleishman, S.J., Whitehead, T.A., Ekiert, D.C. *et al.* (2011) *Science*, **332**, 816–821. https://doi.org/10.1126/science.1202617.

Gao, Z., Tan, C., Chacón, P. *et al.* (2022a) PiFold: Toward effective and efficient protein inverse folding. *arXiv preprint arXiv:2209.12643*. https://doi.org/10.48550/arXiv.2209.12643.

Gao, Z., Tan, C. and Li, S.Z. (2022b) Alphadesign: A graph protein design method and benchmark on alphafolddb. *arXiv preprint arXiv:2202.01079*. https://doi.org/10.48550/arXiv.2202.01079.

Göbel, U., Sander, C., Schneider, R. *et al.* (1994) *Proteins Struct. Funct. Bioinf.*, **18**, 309–317. https://doi.org/10.1002/prot.340180402.

Hsu, C., Verkuil, R., Liu, J. *et al.* (2022) *International Conference on Machine Learning*. PMLR, pp. 8946–8970. https://doi.org/10.1101/2022.04.10.487779.

Huang, P.-S., Boyken, S.E. and Baker, D. (2016) *Nature*, **537**, 320–327. https://doi.org/10.1038/nature19946.

Huang, X., Pearce, R. and Zhang, Y. (2020) *Bioinformatics*, **36**, 1135–1142. https://doi.org/10.1093/bioinformatics/btz740.

Huang, B., Xu, Y., Hu, X. *et al.* (2022) *Nature*, **602**, 523–528. https://doi.org/10.1038/s41586-021-04383-5.

Huang, B., Fan, T., Wang, K. *et al.* (2023) *Bioinformatics*, **39**, btad122. https://doi.org/10.1093/bioinformatics/btad122.

Huszár, F. (2015) How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*. https://doi.org/10.48550/arXiv.1511.05101.

Ingraham, J., Garg, V., Barzilay, R. *et al.* (2019) *Adv. Neural Inf. Process. Syst.*, **32**, 15820–15831.

Ji, S., Xu, W., Yang, M. *et al.* (2012) *IEEE Trans. Pattern Anal. Mach. Intell.*, **35**, 221–231. https://doi.org/10.1109/TPAMI.2012.59.

Jing, B., Eismann, S., Suriana, P. *et al.* (2020) Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*. https://doi.org/10.48550/arXiv.2009.01411.

Johansson, K.E., Johansen, N.T., Christensen, S. *et al.* (2016) *J. Mol. Biol.*, **428**, 4361–4377. https://doi.org/10.1016/j.jmb.2016.09.013.

Jumper, J., Evans, R., Pritzel, A. *et al.* (2021) *Nature*, **596**, 583–589. https://doi.org/10.1038/s41586-021-03819-2.

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

Kuhlman, B. and Bradley, P. (2019) *Nat. Rev. Mol. Cell Biol.*, **20**, 681–697. https://doi.org/10.1038/s41580-019-0163-x.

Kuhlman, B., Dantas, G., Ireton, G.C. *et al.* (2003) *Science*, **302**, 1364–1368. https://doi.org/10.1126/science.1089427.

Leaver-Fay, A., Tyka, M., Lewis, S.M. *et al.* (2011) *Methods Enzymol.*, **487**, 545–574. https://doi.org/10.1016/B978-0-12-381270-4.00019-6.

Li, Z., Yang, Y., Zhan, J. *et al.* (2013) *Annu. Rev. Biophys.*, **42**, 315–335. https://doi.org/10.1146/annurev-biophys-083012-130315.

Li, Z., Yang, Y., Faraggi, E. *et al.* (2014) *Proteins Struct. Funct. Bioinf.*, **82**, 2565–2573. https://doi.org/10.1002/prot.24620.

Li, A.J., Sundar, V., Grigoryan, G. *et al.* (2022) TERMinator: A neural framework for structure-based protein design using tertiary repeating motifs. *arXiv preprint arXiv:2204.13048*. https://doi.org/10.48550/arXiv.2204.13048.

Liang, S., Li, Z., Zhan, J. *et al.* (2022) *Bioinformatics*, **38**, 86–93. https://doi.org/10.1093/bioinformatics/btab598.

Lin,Z. *et al.* (2023) "Evolutionary-scale prediction of atomic-level protein structure with a language model." *Science*, **379.6637**, 1123–1130. https://doi.org/10.1126/science.ade2574.
Liu, R., Wang, J., Xiong, P. *et al.* (2021) *Biotechnol. Bioeng.*, **118**, 2031–2042. https://doi.org/10.1002/bit.27716.
Liu, Y., Zhang, L., Wang, W. *et al.* (2022) *Nat. Comput. Sci.*, **2**, 451–462. https://doi.org/10.1038/s43588-022-00273-6.
Lu, H., Diaz, D.J., Czarnecki, N.J. *et al.* (2022) *Nature*, **604**, 662–667. https://doi.org/10.1038/s41586-022-04599-z.
Mackenzie, C.O., Zhou, J. and Grigoryan, G. (2016) *Proc. Natl. Acad. Sci.*, **113**, E7438–E7447. https://doi.org/10.1073/pnas.1607178113.
Marin, F.I., Johansson, K.E., O'Shea, C. *et al.* (2021) *J. Phys. Chem. B*, **125**, 11141–11149. https://doi.org/10.1021/acs.jpcb.1c05528.
McPartlon, M. and Xu, J. (2023) *Proc. Natl. Acad. Sci.*, **120**, e2216438120. https://doi.org/10.1073/pnas.2216438120.
O'Connell, J., Li, Z., Hanson, J. *et al.* (2018) *Proteins Struct. Funct. Bioinf.*, **86**, 629–633. https://doi.org/10.1002/prot.25489.
Polizzi, N.F. and DeGrado, W.F. (2020) *Science*, **369**, 1227–1233. https://doi.org/10.1126/science.abb8330.
Qi, Y. and Zhang, J.Z. (2020) *J. Chem. Inf. Model.*, **60**, 1245–1252. https://doi.org/10.1021/acs.jcim.0c00043.
Ranzato, M.A., Chopra, S., Auli, M. *et al.* (2015) Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*. https://doi.org/10.48550/arXiv.1511.06732.
Reig, A.J., Pires, M.M., Snyder, R.A. *et al.* (2012) *Nat. Chem.*, **4**, 900–906. https://doi.org/10.1038/nchem.1454.
Ren, M., Yu, C., Bu, D. *et al.* (2023) Highly accurate and robust protein sequence design with CarbonDesign. *bioRxiv*, 2023.2008.2007.552204. https://doi.org/10.1101/2023.08.07.552204.
Rives, A., Meier, J., Sercu, T. *et al.* (2021) *Proc. Natl. Acad. Sci.*, **118**, e2016239118. https://doi.org/10.1073/pnas.2016239118.
Schreier, B., Stumpp, C., Wiesner, S. *et al.* (2009) *Proc. Natl. Acad. Sci.*, **106**, 18491–18496. https://doi.org/10.1073/pnas.0907950106.
Shapovalov, M.V. and Dunbrack, R.L. (2011) *Structure*, **19**, 844–858. https://doi.org/10.1016/j.str.2011.03.019.
Siegel, J.B., Zanghellini, A., Lovick, H.M. *et al.* (2010) *Science*, **329**, 309–313. https://doi.org/10.1126/science.1190239.
Sillitoe, I., Bordin, N., Dawson, N. *et al.* (2021) *Nucleic Acids Res.*, **49**, D266–D273. https://doi.org/10.1093/nar/gkaa1079.
Simonson, T., Gaillard, T., Mignon, D. *et al.* (2013) *J. Comput. Chem.*, **34**, 2472–2484. https://doi.org/10.1002/jcc.23418.
Torng, W. and Altman, R.B. (2017) *BMC Bioinformatics*, **18**, 302–323. https://doi.org/10.1186/s12859-017-1702-0.
Wang, J., Cao, H., Zhang, J.Z. *et al.* (2018) *Sci. Rep.*, **8**, 1–9. https://doi.org/10.1038/s41598-018-24760-x.
Xiong, P., Wang, M., Zhou, X. *et al.* (2014) *Nat. Commun.*, **5**, 5330. https://doi.org/10.1038/ncomms6330.
Xiong, P., Hu, X., Huang, B. *et al.* (2020) *Bioinformatics*, **36**, 136–144. https://doi.org/10.1093/bioinformatics/btz515.
Yan, J., Li, S., Zhang, Y. *et al.* (2023) *Brief. Bioinform.*, **24**, bbad257. https://doi.org/10.1093/bib/bbad257.
Yang, J., Anishchenko, I., Park, H. *et al.* (2020) *Proc. Natl. Acad. Sci.*, **117**, 1496–1503. https://doi.org/10.1073/pnas.1914677117.
Yi, K., Zhou, B., Shen, Y. *et al.* (2023) Graph denoising diffusion for inverse protein folding. *arXiv preprint arXiv:2306.16819*. https://doi.org/10.48550/arXiv.2306.16819.
Zhang, Y. and Skolnick, J. (2004) *Proteins Struct. Funct. Bioinf.*, **57**, 702–710. https://doi.org/10.1002/prot.20264.
Zhang, Y., Chen, Y., Wang, C. *et al.* (2020) *Biophys. J.*, **118**, 43a–44a. https://doi.org/10.1016/j.bpj.2019.11.419.
Zheng, Z., Deng, Y., Xue, D. *et al.* (2023) Structure-informed language models are protein designers. *bioRxiv*, 2023.2002.2003.526917. https://doi.org/10.1101/2023.02.03.526917.
Zhou, X., Chen, G., Ye, J. *et al.* (2023) Protein Sequence Design by Entropy-based Iterative Refinement. *bioRxiv*, 2023.2002.2004.527099. https://doi.org/10.1101/2023.02.04.527099.