
Article:
Bai, P. orcid.org/0000-0003-3027-5518, Miljković, F. orcid.org/0000-0001-5365-505X, Liu, X. orcid.org/0000-0002-3084-519X et al. (4 more authors) (2025) Mask-prior-guided denoising diffusion improves inverse protein folding. Nature Machine Intelligence, 7 (6). pp. 876-888. ISSN 2522-5839

https://doi.org/10.1038/s42256-025-01042-6

---

nature machine intelligence

Article

https://doi.org/10.1038/s42256-025-01042-6

Mask-prior-guided denoising diffusion improves inverse protein folding

Received: 16 November 2024

Accepted: 25 April 2025

Published online: 16 June 2025

Check for updates

Peizhen Bai[1,2], Filip Miljković[3], Xianyuan Liu[1,4], Leonardo De Maria[5], Rebecca Croasdale-Wood[2], Owen Rackham[6] & Haiping Lu[1,4]

Inverse protein folding generates valid amino acid sequences that can fold into a desired protein structure, with recent deep learning advances showing strong potential and competitive performance. However, challenges remain, such as predicting elements with high structural uncertainty, including disordered regions. To tackle such low-confidence residue prediction, we propose a mask-prior-guided denoising diffusion (MapDiff) framework that accurately captures both structural information and residue interactions for inverse protein folding. MapDiff is a discrete diffusion probabilistic model that iteratively generates amino acid sequences with reduced noise, conditioned on a given protein backbone. To incorporate structural information and residue interactions, we have developed a graph-based denoising network with a mask-prior pretraining strategy. Moreover, in the generative process, we combine the denoising diffusion implicit model with Monte-Carlo dropout to reduce uncertainty. Evaluation on four challenging sequence design benchmarks shows that MapDiff substantially outperforms state-of-the-art methods. Furthermore, the in silico sequences generated by MapDiff closely resemble the physico-chemical and structural characteristics of native proteins across different protein families and architectures.

Proteins are complex, three-dimensional (3D) structures folded from linear amino acid (AA) sequences. They play a critical role in essentially all biological processes, including metabolism, immune response and cell cycle control. The inverse protein folding (IPF) problem is a fundamental structure-based protein design problem in computational biology and medicine. It aims to generate valid AA sequences with the potential to fold into a desired 3D backbone structure, enabling the creation of new proteins with specific functions[1]. Its enormous applications range from therapeutic protein engineering, lead compound optimization and antibody design[2].

Traditional physics-based approaches consider IPF as an energy optimization problem[3], suffering from high computational cost and limited accuracy. In recent years, deep learning has emerged as the preferred paradigm for solving protein-structure problems owing to its strong ability to learn complex nonlinear patterns from data adaptively. In deep learning for IPF, early convolutional neural network-based models view each protein residue as an isolated unit or the whole as point cloud data, with limited consideration of structural information and interactions between residues[4–7]. Recently, graph-based methods have represented 3D protein structures as proximity graphs, and then use graph neural networks (GNNs) to model residue representations and incorporate structural constraints. GNNs can aggregate and exchange local information within graph-structured data, enabling substantial performance improvement in graph-based methods.

[1]School of Computer Science, University of Sheffield, Sheffield, UK. [2]Biologics Engineering, Oncology R&D, AstraZeneca, Cambridge, UK. [3]Medicinal Chemistry, Research and Early Development, Cardiovascular, Renal and Metabolism, BioPharmaceuticals R&D, AstraZeneca, Gothenburg, Sweden. [4]Centre for Machine Intelligence, University of Sheffield, Sheffield, UK. [5]Medicinal Chemistry, Research and Early Development, Respiratory and Immunology, BioPharmaceuticals R&D, AstraZeneca, Gothenburg, Sweden. [6]School of Biological Sciences, University of Southampton, Southampton, UK. e-mail: h.lu@sheffield.ac.uk

https://doi.org/10.1038/s42256-025-01042-6

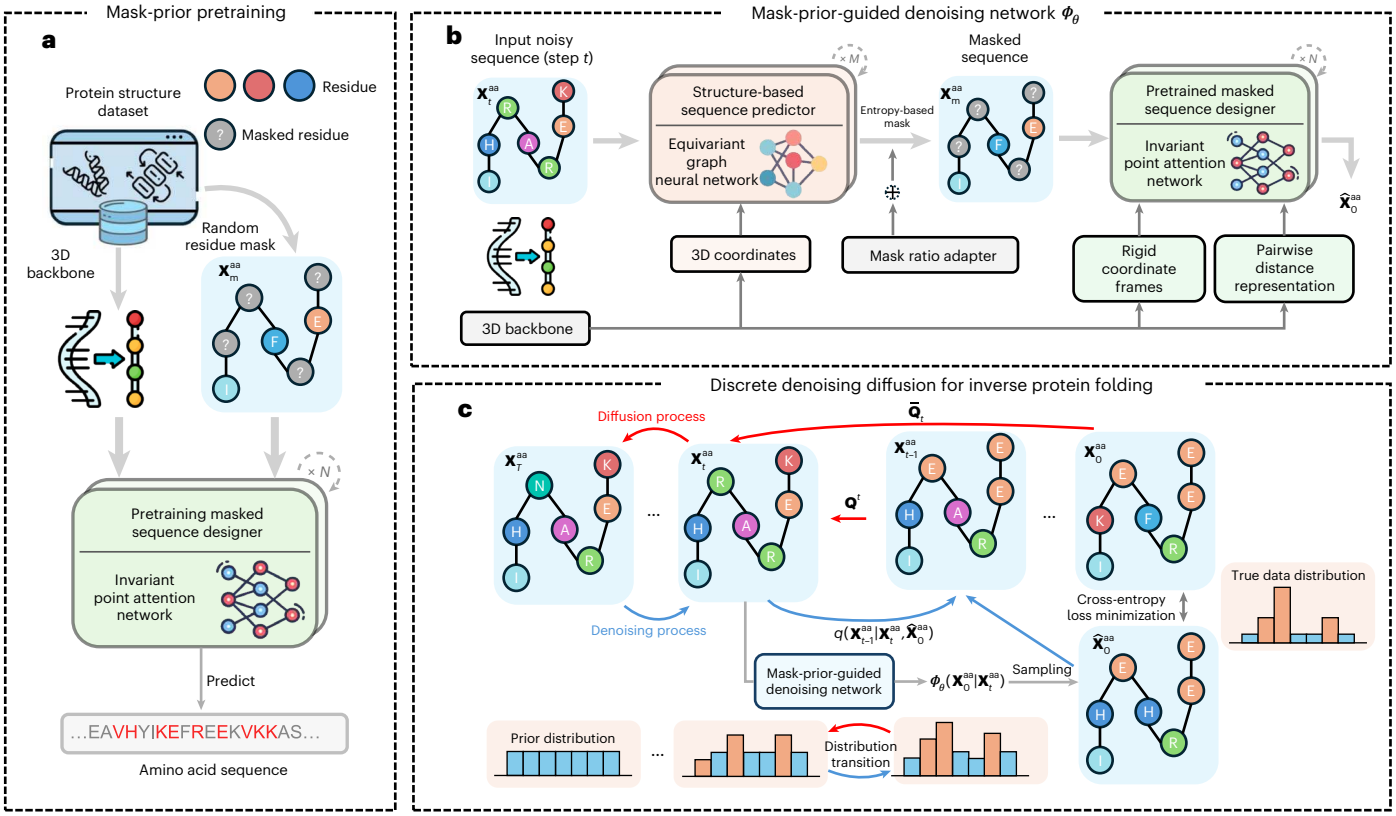

Fig. 1 | Mask-prior-guided denoising diffusion (MapDiff) for inverse protein folding. a, The mask-prior pretraining stage randomly masks residues within the AA sequence and pretrains an invariant point attention (IPA) network with the masked sequence and the 3D backbone structure to learn prior structural and sequence knowledge, using BERT-like masked language modelling objectives. **b**, The mask-prior-guided denoising network ϕ_θ takes an input noisy AA sequence $\mathbf{X}^{aa}$ to predict the native AA sequence $\mathbf{X}_0^{aa}$ by means of three operations in every iterative denoising step. It first initializes a structure-based sequence predictor as an equivariant graph neural network to denoise the noisy sequence $\mathbf{X}^{aa}$ conditioned on the provided 3D backbone structure. Then, combining an entropy-based mask strategy with a mask ratio adaptor identifies and masks low-confidence residues in the denoised sequence in the first step to produce a masked sequence $\mathbf{X}_m^{aa}$. Next, the pretrained masked sequence designer in **a** takes the masked sequence $\mathbf{X}_m^{aa}$ and its 3D backbone information for refinement (fine-tuning) to better predict the native sequence $\mathbf{X}_0^{aa}$. **c**, The MapDiff denoising diffusion framework iteratively alternates between two processes: diffusion and denoising. The diffusion process progressively adds random discrete noise to the native sequence $\mathbf{X}_0^{aa}$ according to the cumulative transition matrix $\bar{\mathbf{Q}}_t$ at the diffusion step t so that the real data distribution can gradually transition to a uniform or marginal prior distribution. The denoising process randomly samples an initial noisy AA sequence $\mathbf{X}_T^{aa}$ from the prior distribution and iteratively uses the denoising network ϕ_θ in **b** to denoise it, learning to predict the native sequence $\mathbf{X}_0^{aa}$ from $\mathbf{X}_t^{aa}$ at each denoising step t. The prediction $\hat{\mathbf{X}}_0^{aa}$ facilitates the computation of the posterior distribution $q(\mathbf{X}_{t-1}^{aa}|\mathbf{X}_t^{aa}, \hat{\mathbf{X}}_0^{aa})$ for predicting a less-noisy sequence $\mathbf{X}_{t-1}^{aa}$.

Despite the advances in graph-based methods, structural information alone cannot determine the residue identities of some challenging structural elements, such as intrinsically disordered regions[8]. In such uncertain, low-confidence cases, interactions with other accurately predicted residues can provide more reliable guidance for mitigating uncertainty in these regions. Moreover, existing deep learning-based IPF methods typically employ autoregressive decoding or uniformly random decoding to generate AA sequences, are prone to accumulating prediction errors[9,10] and are limited in capturing global and long-range dependencies in protein evolution[11,12]. Recently, several non-autoregressive alternatives have shown the potential to outperform the autoregressive paradigm in related contexts[9,13,14]. In addition, protein-structure prediction methods, such as the AlphaFold series[15,16], often take an iterative generation process to refine non-deterministic structures by integrating well-predicted information. These raise the question: can combining residue interactions with an iterative refinement and an efficient non-autoregressive decoding improve IPF prediction performance to generate more plausible protein sequences?

Recently, denoising diffusion models, an innovative class of deep generative models, have gained growing attention in various fields. They learn to generate conditional or unconditional data by iteratively denoising random samples from a prior distribution. Diffusion-based models have been adopted for de novo protein design and molecule generation, achieving state-of-the-art performance. For example, RFdiffusion[17] fine-tunes the protein structure prediction network RoseTTAFold[18] under a denoising diffusion framework to generate 3D protein backbones, and torsional diffusion[19] implements a diffusion process on the space of torsion angles for molecular conformer generation. In structure-based drug design, DiffSBDD[20] proposes an equivariant 3D-conditional diffusion model to generate new small-molecule binders conditioned on target protein pockets. Although diffusion models have a widespread application in computational biology, most existing methods focus primarily on generating structures in continuous 3D space. The potential of diffusion models in inverse folding has not yet been fully exploited.

We propose a mask-prior-guided denoising diffusion (MapDiff) framework (Fig. 1) to accurately capture structure-to-sequence mapping for IPF prediction. Unlike previous graph-based methods, MapDiff models IPF as a discrete denoising diffusion problem that iteratively generates less-noisy AA sequences conditioned on a target protein structure. Owing to the property of denoising diffusion, MapDiff can

https://doi.org/10.1038/s42256-025-01042-6

Table 1 | Performance comparison on the CATH 4.2 and CATH 4.3 datasets with topology classification split

	Models	External knowledge	Model parameters	Perplexity (↓)			Median recovery rate (%, ↑)		
				Short	Single-chain	Full	Short	Single-chain	Full
CATH 4.2	[a]StructGNN[26]	✗	1.4M	8.29	8.74	6.40	29.44	28.26	35.91
	[a]GraphTrans[26]	✗	1.5M	8.39	8.83	6.63	28.14	28.46	35.82
	[a]GVP[43]	✗	2.0M	7.09	7.49	6.05	32.62	31.10	37.64
	[a]AlphaDesign[44]	✓	6.6M	7.32	7.63	6.30	34.16	32.66	41.31
	ProteinMPNN[1]	✗	1.9M	6.90	7.03	4.70	36.45	35.29	48.63
	PiFold[13]	✗	6.6M	5.97	*6.13*	4.61	39.17	42.43	51.40
	LM-Design[45]	✓	659M	6.86	6.82	4.55	37.66	38.94	*53.19*
	GRADE-IF[38]	✗	7.0M	*5.65*	6.46	*4.40*	*45.84*	*42.73*	52.63
	MapDiff (uniform prior)	✗	14.7M	3.99	4.43	3.46	52.85	**50.00**	**61.03**
	MapDiff (marginal prior)	✗	14.7M	**3.96**	**4.41**	**3.43**	**54.04**	49.34	60.93
CATH 4.3	[a]GVP-GNN-Large[27]	✗	21M	7.68	6.12	6.17	32.60	39.40	39.20
	[a]+ AF2 predicted data	✓	142M	6.11	4.09	4.08	38.30	50.08	50.08
	[a]GVP-Transformer[27]	✗	21M	8.18	6.33	6.44	31.30	38.50	38.30
	[a]+ AF2 predicted data	✓	142M	6.05	4.00	4.01	38.10	51.50	51.60
	ProteinMPNN[1]	✗	1.9M	6.12	6.18	4.63	40.00	39.13	47.66
	PiFold[13]	✗	6.6M	5.52	*5.00*	*4.38*	43.06	45.54	51.45
	LM-Design[45]	✓	659M	6.01	5.73	4.47	44.44	45.31	*53.66*
	GRADE-IF[38]	✗	7.0M	*5.30*	6.05	4.58	*48.21*	*45.94*	52.24
	MapDiff (uniform prior)	✗	14.7M	**3.88**	3.85	**3.48**	**55.95**	54.65	**60.86**
	MapDiff (marginal prior)	✗	14.7M	3.90	**3.83**	3.52	55.56	**54.99**	60.68

The results include the perplexity and median recovery rate on the full test set, as well as on short and single-chain subsets. The external knowledge column indicates whether additional training data or protein language models are used. [a]We also quote partial baseline results from ref. 13 and ref. 27 for comparative analysis. The best result for each dataset and metric is marked in **bold** and the second-best result is in *italics*.

also be viewed as an iterative refinement that enhances the accuracy of the generated sequences over time. Moreover, we have designed a two-step denoising network to adaptively improve the denoising trajectories using a pretrained mask prior. Our denoising network effectively leverages the structural information and residue interactions to reduce prediction error on low-confidence residue prediction. To further improve the denoising speed and uncertainty estimation, we combine the DDIM[21] with Monte-Carlo dropout[22] in the discrete generative process. DDIM accelerates sequence generation by skipping multiple denoising steps, whereas Monte-Carlo dropout reduces uncertainty by performing multiple stochastic forward passes with dropout enabled during inference. We conducted performance comparisons against state-of-the-art methods for IPF prediction, demonstrating the effectiveness of MapDiff across multiple metrics and benchmarks, outperforming even those incorporating external knowledge. Moreover, when we used AlphaFold2[15] to fold the sequences generated by MapDiff back to 3D structures, such AlphaFold2-folded structures were highly similar to the native protein templates, even for cases of low sequence recovery rates.

This work shows the high potential of using discrete denoising diffusion models with mask-prior pretraining for IPF prediction. Our main contributions are three-fold: (1) we propose a discrete denoising diffusion-based framework named MapDiff to explicitly consider the structural information and residue interactions in the diffusion and denoising processes; (2) we have designed a mask-prior-guided denoising network that adaptively denoises the diffusion trajectories to produce feasible and diverse sequences from a fixed structure; and (3) MapDiff incorporates discrete DDIM with Monte-Carlo dropout to accelerate the generative process and improve uncertainty estimation.

Results

MapDiff framework

As shown in Fig. 1, the MapDiff framework formulates IPF prediction as a denoising diffusion problem (Fig. 1c). The diffusion process progressively adds random discrete noise to the native AA sequence according to the transition probability matrices to facilitate the training of a denoising network. In the denoising process, this denoising network iteratively denoises a noisy, randomly sampled AA sequence conditioned on the 3D structural information to predict or reconstruct the native AA sequence. The diffusion and denoising processes iterate alternately to capture the sampling diversity of native sequences from their complex distribution and refine the predicted AA sequences.

We propose a mask-prior-guided denoising network to adaptively adjust the discrete denoising trajectories towards generating more valid AA sequences by means of three operations within each iterative denoising step (Fig. 1b). First, a structure-based sequence predictor employs an equivariant graph neural network (EGNN)[23] to denoise the noisy sequence conditioned on the backbone structure. Second, we use an entropy-based mask strategy[24] and a mask ratio adaptor to identify and mask low-confidence or uncertain (for example, structurally undetermined) residues in the denoised sequence in the first operation to produce a masked sequence. Third, a pretrained masked sequence designer network predicts the masked residues to obtain their refined prediction. The pretraining of the masked sequence designer is done before the diffusion and denoising processes by means of an invariant point attention (IPA) network[15] using masked language modelling (Fig. 1a), incorporating prior structural and sequence knowledge. The structure-based sequence predictor and masked sequence designer refine denoising trajectories by leveraging structural information and residue interactions. For efficient sequence generation, the denoising

https://doi.org/10.1038/s42256-025-01042-6

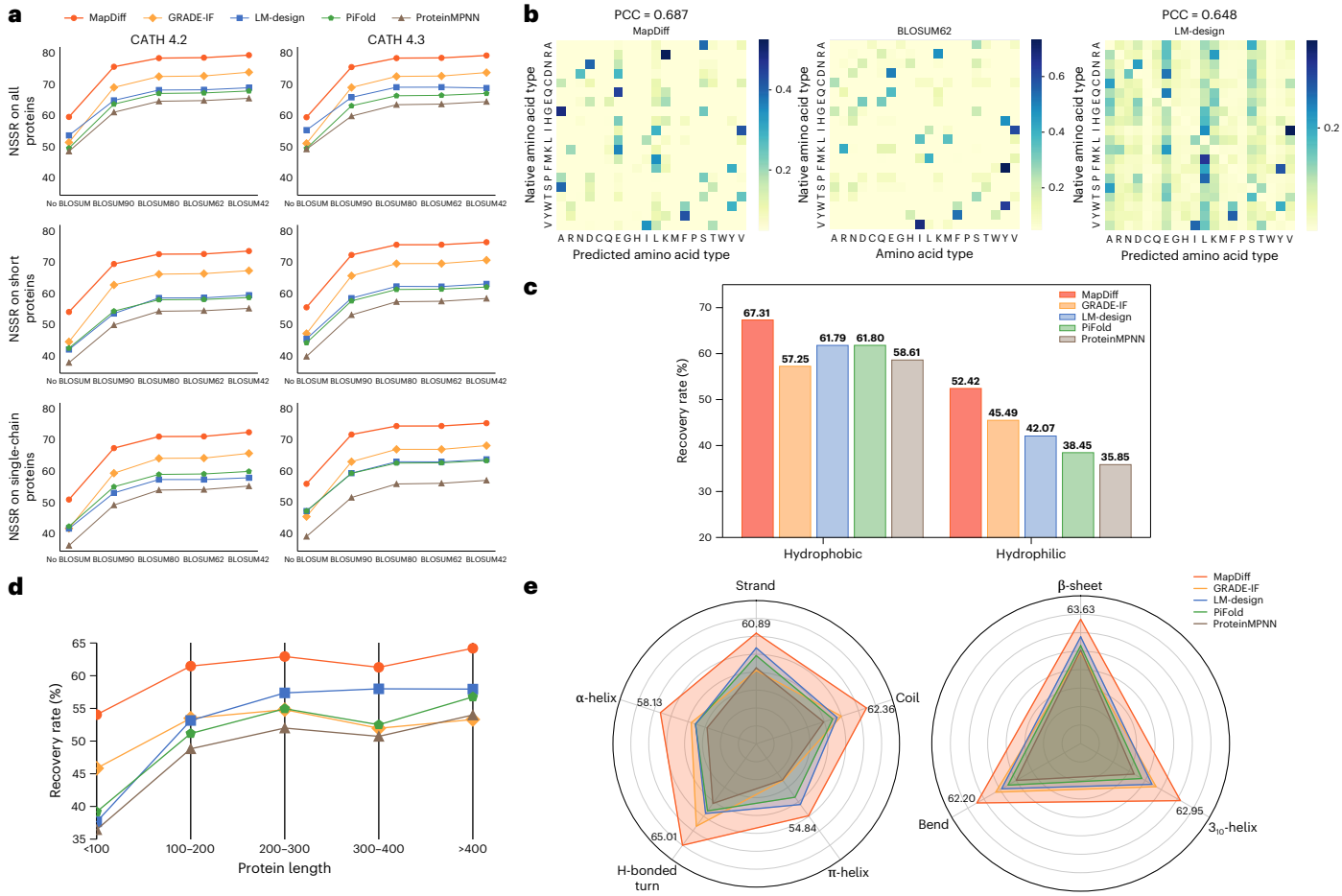

Fig. 2 | Model performance comparison and sensitivity analysis across different scenarios on the CATH datasets. a, NSSR scores for MapDiff and baseline methods[1,13,38,45] on the full test sets and the short and single-chain protein subsets for four different BLOSUM matrices and no BLOSUM matrix. **b**, Sum-normalized confusion matrices for MapDiff (left) and LM-Design (right) predictions, and the softmax-normalized native BLOSUM62 matrix (middle). Darker colours indicate a higher predicted likelihood, with diagonal elements masked. The Pearson correlation coefficient (PCC) between the predicted matrix and BLOSUM62 quantifies their similarity and linear correlation. **c**, Breakdown of the recovery rates into hydrophilic and hydrophobic residues. **d**, Median sequence recovery rates across different protein lengths. **e**, Residue recovery performance across different secondary structures visualized in two groups for clarity, defined using the DSSP algorithm[57]. Coils correspond to regions without regular secondary structures and are considered disordered regions. Bends and H-bonded turns are regarded as less ordered regions owing to their flexibility and transient nature. For these regions, MapDiff outperforms the baselines substantially.

network uses non-autoregressive decoding to generate sequences in a one-shot manner[13]. In addition, we incorporate DDIM[21] to accelerate inference by skipping multiple denoising steps and Monte-Carlo dropout[22] to reduce uncertainty. The Methods provides more details.

Evaluation strategies and metrics

We conducted experiments across diverse datasets to evaluate MapDiff against state-of-the-art protein sequence design methods. We first evaluated two popular benchmark datasets, CATH 4.2 and CATH 4.3 (ref. 25), using the same topology-based data split employed in previous works[13,26,27]. In addition to the full test sets, we also studied two subcategories of generated proteins: short proteins up to 100 residues in length and single-chain proteins (labelled with one chain in CATH). We used another two distinct datasets, TS50 (ref. 5) and PDB2022 (ref. 24) to evaluate the zero-shot generalization of models. Furthermore, we studied the foldability of the generated protein sequences by means of AlphaFold2 (ref. 15) by comparing the discrepancy between the AlphaFold2-refolded structures and ground-truth native structures. This is an in silico evaluation rather than definitive proof that the designed sequences can fold into their intended structures. The 'Experimental setting' section provides detailed information and statistics about these datasets.

We evaluated the accuracy of generated sequences using three metrics: perplexity, recovery rate and native sequence similarity recovery (NSSR)[28]. Perplexity measures the alignment between a model's predicted AA probabilities and the native AA types at each residue position. The recovery rate indicates the proportion of accurately predicted AAs in the protein sequence. The NSSR evaluates the similarity between the predicted and native residues by means of the blocks substitution matrix (BLOSUM)[29], where each residue pair contributes to a positive prediction if their BLOSUM score is greater than zero. We used BLOSUM42, BLOSUM62, BLOSUM80 and BLOSUM90 to account for AA similarities at four different cutoff levels for NSSR computation. To evaluate the foldability, that is, the quality of refolded protein structures, we used six metrics: predicted local distance difference test (pLDDT), predicted aligned error (PAE), predicted template modelling (pTM), template modelling score (TM-score), root mean square deviation (RMSD) and global distance test-total score (GDT-TS), where pLDDT, PAE and pTM measure the confidence and reliability of predicted structures produced by AlphaFold2, and TM-score, RMSD and GDT-TS measure the discrepancies between the predicted 3D structures and their native counterparts. Supplementary Information Section 9 provides the technical details for these metrics.

https://doi.org/10.1038/s42256-025-01042-6

Table 2 | Transferability: zero-shot performance comparison on transferability from CATH to PDB2022 and TS50 datasets

	Models	PDB2022			TS50		
		Recovery (↑)	NSSR62 (↑)	NSSR90 (↑)	Recovery (↑)	NSSR62 (↑)	NSSR90 (↑)
Transferability	ProteinMPNN[1]	56.75 (56.65)	72.50 (72.59)	69.96 (69.95)	52.34 (51.80)	70.31 (70.13)	66.77 (66.80)
	PiFold[13]	60.63 (60.26)	75.55 (75.30)	72.96 (72.86)	*58.39* (58.90)	73.55 (74.52)	70.33 (71.33)
	LM-Design[45]	*66.03 (66.20)*	*79.55 (80.12)*	*77.60 (78.20)*	57.62 (58.27)	73.74 (75.69)	71.22 (73.12)
	GRADE-IF[38]	58.09 (58.35)	77.44 (77.51)	74.57 (74.97)	57.74 (*59.27*)	*77.77 (79.11)*	*74.36 (76.24)*
	MapDiff	**68.03 (68.00)**	**84.19 (84.30)**	**82.13 (82.29)**	**68.76 (69.77)**	**84.10 (85.27)**	**81.76 (83.08)**
	Models	**CATH 4.2**					
		pLDDT (↑)	**PAE (↓)**	**PTM (↑)**	**TM-score (↑)**	**RMSD (↓)**	**GDT-TS(↑)**
Foldability	ProteinMPNN[1]	87.13±9.79	5.85±3.17	77.42±14.96	86.27±16.32	*3.08±4.25*	*85.08±15.53*
	PiFold[13]	87.42±9.82	5.81±3.22	77.75±15.03	*86.56±16.21*	3.10±4.29	85.47±15.49
	LM-Design[45]	*88.04±9.00*	*5.78±3.27*	*78.00±14.80*	85.36±16.98	3.54±5.00	84.08±16.45
	GRADE-IF[38]	85.32±9.27	6.30±3.10	75.63±13.8 0	85.80±14.93	3.11±3.96	83.37±14.43
	MapDiff	**88.63±8.27**	**5.42±2.76**	**79.00±13.04**	**88.77±13.48**	**2.57±3.50**	**87.75±13.24**

We report the test results of models when trained on CATH 4.2 and CATH 4.3, with the results for CATH 4.3 in parentheses. Foldability: foldability comparison for the generated sequences on the CATH 4.2 test set using AlphaFold2. The results are presented as mean±standard deviation. The best result for each dataset and metric is marked in **bold** and the second-best result is in *italics*.

Sequence recovery performance

First, we evaluated MapDiff's sequence recovery with uniform or marginal priors against state-of-the-art baselines on the CATH datasets. Table 1 presents the prediction perplexity and median recovery rate on the full test set, along with short and single-chain subsets. The results demonstrate that MapDiff achieves the best performance across different metrics and subsets of data, highlighting its effectiveness in generating valid protein sequences. Specifically, we observe that: (1) MapDiff achieves a recovery rate of 61.03% and 60.86% on the full CATH 4.2 and CATH 4.3 test sets, substantially outperforming the baselines by 7.74% and 7.20%, respectively. Furthermore, MapDiff shows recovery improvements of 8.20% and 6.61% on the short and single-chain test sets of CATH 4.2.; (2) MapDiff consistently achieves the lowest perplexity compared with previous methods and produce high-confidence probability distribution to facilitate accurate predictions; (3) MapDiff is a highly accurate IPF model that operates independently of external knowledge. In some of the compared baselines, external knowledge sources, such as additional training data or protein language models, are used to enhance prediction accuracy. Owing to its well-designed architecture and diffusion-based generation mechanism, MapDiff effectively uses limited training data to capture relevant patterns to achieve superior generalizability; and (4) MapDiff's performance is largely unaffected by the choice of prior distribution. Therefore, we use the marginal prior[30] in our experiments, as it is data-driven and better aligns with the true amino acid distribution.

We further study model performance across different scenarios. Figure 2a presents the mean NSSR scores for MapDiff and the baselines on the CATH datasets. MapDiff consistently achieves the best NSSR scores across different test sets. Figure 2b compares the confusion matrices of MapDiff and LM-Design with the native BLOSUM62 matrix on CATH 4.2. For clearer visualization and comparison, we normalized these matrices to the [0,1] probability range, with the diagonal elements masked. The confusion matrix denotes proportions for specific combinations of actual and predicted amino acid types, with darker cells indicating greater proportion. Many non-diagonal darker cells in MapDiff highlight the alignment between closely related residue pairs, as defined by the BLOSUM62 matrix, indicating that MapDiff can effectively capture the homologous substitutions between residues. In addition, MapDiff's higher correlation with BLOSUM62 than LM-Design suggests a stronger alignment with substitution preferences.

Figure 2c,e shows the sequence recovery performance across different amino acid types, as well as eight secondary structures. Notably, MapDiff is the only model achieving over 50% recovery rate in predicting hydrophobic amino acids and substantial improvements in recovering α-helix and β-sheet secondary structures. Figure 2d presents a sensitivity analysis of the recovery performance for varying protein lengths. For short proteins (less than 100 amino acids in length), several baselines show a marked decrease in performance. For example, the recovery rate of LM-Design falls below 40% for the short proteins. This could be due to the protein language model used in LM-Design being sensitive to protein length. By contrast, MapDiff, which employs a mask-prior-guided denoising network and an iterative denoising process, consistently outperforms all baselines and maintains high performance across all protein lengths.

To validate the zero-shot transferability of our method, we compared the model's performance on two independent test datasets, TS50 and PDB2022, which do not overlap with the CATH data, as shown in Table 2. The results demonstrate that MapDiff achieves the highest recovery and NSSR scores on both datasets. We can conclude that, even though LM-Design reaches a high recovery (66%) that is approaching our method on PDB2022, the performance gap widens on NSSR62 and NSSR90. By contrast, GRADE-IF and MapDiff can generalize better when considering the possibility of similar residue substitution. This suggests that diffusion-based models more effectively capture residue similarity in IPF prediction. For the TS50 dataset, MapDiff substantially improves state-of-the-art methods by 6.33% on NSSR62, and is the best model, achieving a recovery rate of 68%.

Foldability of generated protein sequences

Foldability is a crucial property that evaluates whether a protein sequence can fold into the desired structure. In this study, we evaluated the foldability of generated protein sequences by predicting their structures with AlphaFold2 and comparing the discrepancies against the native crystal structures. Table 2 presents six foldability metrics for the 1,120 structures in the CATH 4.2 test set. The results indicate that the generated protein sequences by MapDiff exhibit superior foldability, the highest confidence and minimal discrepancy compared with their native structures. Notably, the foldability and sequence recovery

https://doi.org/10.1038/s42256-025-01042-6

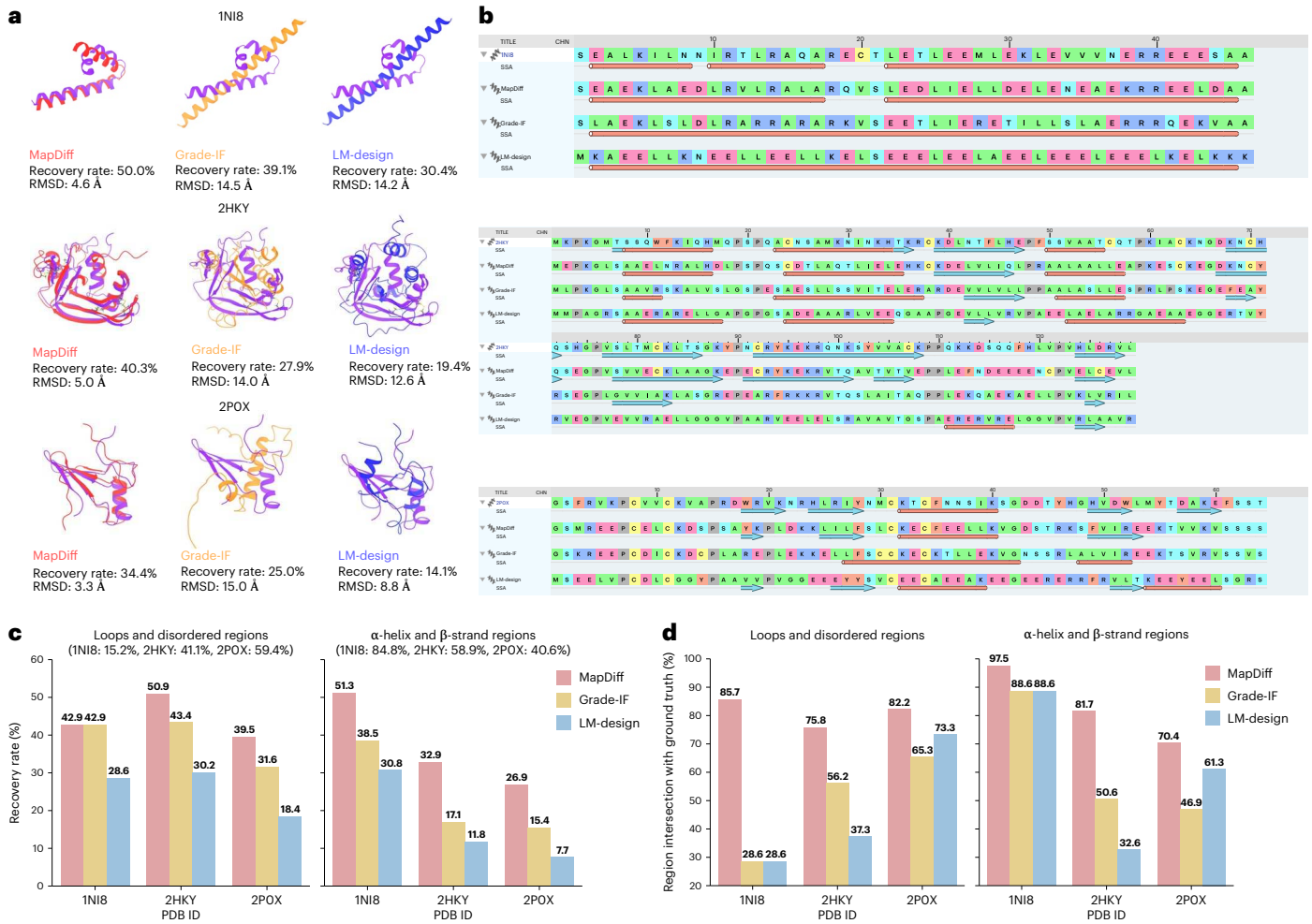

Fig. 3 | Comparison of three refolded structures (left) and the respective model-designed sequences (right) for proteins with PDB IDs 1NI8, 2HKY and 2P0X. a, Refolded tertiary structure visualization of the sequences designed by three models MapDiff (red), GRADE-IF (orange) and LM-Design (blue). The refolded structures were generated by AlphaFold2 and superposed against the ground-truth structures (purple). For each model and structure, the recovery rate and RMSD value are indicated for foldability comparison. **b**, The alignment of the three native sequences and the respective model-designed sequences. The results are shown with secondary structure elements marked below each sequence: α-helices are shown in red cylinders, β-strands in blue arrows, and loops and disordered regions are unmarked. For the native proteins, the secondary structures were derived from their source PDB files. For the predicted proteins, the secondary structures were assigned by first identifying all interbackbone hydrogen bonds and then searching for hydrogen-bonding patterns that represent helices and strands. The refolded structures and sequence alignments are visualized using the Schrödinger Maestro software[58]. **c**, Recovery rates for loops and disordered regions (left panel) and α-helix and β-strand regions (right panel) across three structures. Bars indicate the recovery rates of three methods (MapDiff, Grade-IF and LM-Design). The percentage composition of regions for each structure is provided below the panel titles. MapDiff consistently achieves the highest recovery rates across different categories of regions for the three structures, with an average improvement of 5.1% in loops and disordered regions and 13.4% in α-helix and β-strand regions compared with Grade-IF. **d**, Jaccard region intersections between the predicted and ground-truth structures for loops and disordered regions (left panel) versus α-helix and β-strand regions (right panel). The Jaccard index measures the fraction of the overlap between two sets, and the results demonstrate that MapDiff achieves the highest score across both categories of regions.

results do not always positively correlate. For example, although ProteinMPNN performs poorly in sequence recovery, it achieves the best RMSD among baseline methods. Therefore, it is essential to comprehensively evaluate IPF models from both sequence and structure perspectives. Supplementary Information Section 2 and Supplementary Fig. 2 present analysis of the right-skewed RMSD distribution[31].

In Fig. 3a, we illustrate exemplary 3D structures refolded by AlphaFold2 from IPF-derived sequences generated by MapDiff, GRADE-IF and LM-Design for three different protein folds (PDB ID 1NI8 (ref. 32), 2HKY (ref. 33) and 2P0X (ref. 34) with a preselected monomer pTM prediction argument. In addition to estimating the sequence recovery rate and foldability of derived 3D structures using the RMSD metric, we also inspected the alignment of native and generated sequences, including the agreement between refolded secondary structures and individual pairs of amino acids in Fig. 3b. Figure 3c,d presents quantitative analyses of performance on different regions.

The first example is a 46-amino-acid-long monomer of the 1NI8 structure (purple) representing an amino-terminal (N-terminal) fragment of the H-NS dimerization domain, a protein composed of three α-helices that is involved in structuring the chromosome of Gram-negative bacteria, and hence acts as a global regulator for the expression of different genes[32]. Two monomers form a homodimer which requires the presence of K5, R11, R14, R18 and K31 residues to engage in the prokaryotic DNA binding. MapDiff (red) managed to retrieve two out of the three α-helices, with an interhelical turn present at the same position as in the original structure (A17-R18), whereas GRADE-IF (orange) and LM-Design (blue) models only consisted of a single continuous α-helix. Moreover, MapDiff and GRADE-IF obtained

Table 3 | Ablation study of the denoising network modules in MapDiff

Module	Component	MapDiff	Variant 1	Variant 2	Variant 3	Variant 4	Variant 5
G-EGNN	EdgeUpdate	✓	✓	✓		✓	✓
	CoordinateUpdate	✓	✓	✓			✓
	GlobalContext	✓	✓		✓	✓	✓
Refinement	MaskAdaptor	✓		✓	✓	✓	
	IPA network	✓	✓	✓	✓	✓	
Sequence	Recovery (↑, %)	**60.93**	58.64	59.76	58.38	60.16	56.46
	NSSR62 (↑, %)	**78.57**	76.73	77.32	77.04	77.80	75.69
	NSSR90 (↑, %)	**75.66**	73.52	74.82	74.24	75.02	72.58
Foldability	pLDDT (↑)	**88.63**	88.08	88.24	87.95	88.30	86.95
	PTM (↑)	**79.00**	78.42	78.61	78.24	78.74	77.20
	PAE (↓)	**5.42**	5.57	5.54	5.62	5.49	5.86
	TM-Score (↑, %)	**88.77**	88.25	88.47	88.16	88.58	87.50
	GDT-TS (↑, %)	**87.75**	87.12	87.40	86.96	87.53	85.72
	RMSD (↓)	**2.57**	2.67	2.65	2.65	2.53	2.76
Summary	Change	-	↓↓	↓	↓↓	↓	↓↓↓

We studied five model variants and investigated how much sequence recovery and foldability metrics decreased when key components were removed on CATH 4.2. The best result for each metric is marked in **bold**.

four out of five (K5, R11, R14 and R18) amino acids required for DNA binding and LM-Design obtained none. MapDiff and LM-Design generate glutamic acid (E) and GRADE-IF isoleucine (I), which, in comparison with the corresponding positively charged K31 in the original structure, are negatively charged and neutral residues, respectively. The single continuous α-helix displayed by GRADE-IF and LM-Design AlphaFold2 models hence produces much worse RMSD values (14.5 Å and 14.2 Å, respectively) than the MapDiff model, which retrieved two helices at the right positions (RMSD = 4.6 Å). Consistent with this, MapDiff obtained a 10% higher recovery rate than GRADE-IF and LM-Design.

The second example is the 2HKY structure of 109-amino-acid-long human ubiquitous ribonuclease 7 (hRNase7), rich in positively charged residues, that possesses antimicrobial activity[33]. This α/β mixed protein contains 22 cationic residues (18 K and 4 R) distributed into three surface-exposed clusters that promote binding to the bacterial membrane, which thus renders it permeable, which consequently elicits membrane disruption and death. In addition, it contains four disulfide bridges (C24–C82, C38–C92, C56–C107 and C63–C70), which are critical for its secondary and tertiary structure, three of which were successfully retrieved by MapDiff, whereas no cysteines were found in either GRADE-IF or LM-Design sequences. Furthermore, all secondary structure elements were nearly entirely recovered by MapDiff, unlike GRADE-IF and LM-Design solutions which contained little resemblance to the native structure, particularly in the carboxy-terminus (C-terminus) half. These structural findings were reflected in a fair recovery rate of 40.3% and an RMSD value of 5.0 Å for MapDiff, which was considerably better than in GRADE-IF and LM-Design structures (14.0 Å and 12.6 Å, respectively).

A third example displays AlphaFold2-refolded structures obtained from generated sequences with relatively low recovery rates that used the 2P0X structure of an optimized non-biological (de novo) ATP-binding protein as a template[34]. Here MapDiff retrieved all detected secondary structure elements, except for the C-terminus β-strand which was replaced by a loop. LM-Design was the second best with an α-helix substituting the aforementioned β-strand. Even if nearly all secondary structure elements were retrieved by both MapDiff and LM-Design AlphaFold2 models, the MapDiff model obtained by far the best RMSD (3.3 Å as opposed to 8.8 Å). Despite having a better recovery rate than LM-Design, GRADE-IF generated a sequence that folded poorly compared with the experimentally confirmed structure (15.0 Å).

In these cases, MapDiff achieved low RMSD values to successfully replicate the majority of secondary structure elements elucidated through experiments, including other structural features such as the disulfide bonds (2HKY) or positively charged residues that were suspected to participate in protein function (1NI8). By contrast, Grade-IF and LM-Design predicted sequences that not only had lower recovery rates than MapDiff but also exhibited partially or entirely absent secondary structure elements, as shown by the experimentally derived 3D structures, resulting in substantially worse RMSDs. Although the structures predicted by AlphaFold2 cannot entirely substitute the structural elucidation by experimental techniques such as X-ray or NMR (nuclear magnetic resonance), they provide the first glance at the foldability potential of de novo generated protein sequences by IPF models. A natural next step in future work would be to express the de novo designed protein sequences and experimentally determine their tertiary structures.

Supplementary Information Section 1 and Supplementary Fig. 1 study the closest training structures and sequences of the three examples. The highest TM-scores for 1NI8, 2HKY and 2P0X from structures in the training set were 0.57 (1A7W), 0.25 (1V88) and 0.33 (1WIM), respectively, indicating that there are no highly similar structures during training. Similarly, the highest BLAST[35] bit-scores for sequences in the training set were 23.1 (4ZEO), 26.6 (2BM8) and 24.6 (3MSR), respectively, indicating that no highly similar sequences are present during training.

Model analysis and ablation study

We performed analysis and ablation studies to assess the effectiveness of key components in MapDiff. We focused on investigating the contributions of edge feature updating, node coordinate updating and global context learning within the base sequence predictor (G-EGNN) to the model performance. In addition, we examined the impact of the mask ratio adaptor and the pretrained IPA network in the residue refinement module on the predictions. As shown in Table 3, we studied five variants of MapDiff, each with different key components removed, and compared their results with the CATH 4.2 test set. The results show that each component positively enhanced the sequence recovery and

foldability performance. For example, the IPA-based refinement mechanism (variant 5) achieved the most substantial sequence improvement, increasing recovery by 4.47%, whereas the global context learning and coordinate updating (variants 2 and 4) in G-EGNN improved the recovery by 1.17% and 0.77%, respectively. The impact on foldability increases with sequence recovery performance but remains less pronounced, indicating that AlphaFold2 is robust to these variations and predicts stable protein folds. In addition, Supplementary Information Section 7 and Supplementary Fig. 3 analyse MapDiff's sensitivity to the number of Monte-Carlo samples and DDIM skipping steps.

Discussion

In this work, we present MapDiff, a mask-prior-guided denoising diffusion framework for structure-based protein design. Specifically, we regard IPF prediction as a discrete denoising diffusion problem, and developed a graph-based denoising network to capture structural information and residue interactions. At each denoising step, we used a G-EGNN module to generate clean sequences from input structures and a pretrained IPA module to refine low-confidence residues, ensuring reliable denoising trajectories. Moreover, we integrated DDIM with Monte-Carlo dropout to accelerate generative sampling and enhance uncertainty estimation. Experiments demonstrate that MapDiff consistently outperforms the state-of-the-art IPF models across multiple benchmarks and scenarios. At the same time, the generated protein sequences exhibit a high degree of similarity to their native counterparts. Even in cases where the overall sequence similarity was low, these sequences could often refold into their native structures, as demonstrated by the AlphaFold2-refolded models. We also conducted a comprehensive ablation study to analyse the importance of different model components for the prediction results. MapDiff demonstrates transferability and robustness in generating new protein sequences, even with limited training data. Promising future directions include verifying the applicability of MapDiff in practical domains such as de novo antibody design and protein engineering, incorporating predicted structures from structure prediction models as external data for incremental training, integrating physics-informed constraints, leveraging sequential evolutionary knowledge from protein language models to further refine residue predictions, and further validating the foldability of the designed sequences by conducting folding simulations or molecular dynamics simulations.

Methods

Discrete denoising diffusion models

Denoising diffusion models are a class of deep generative models trained to create new samples by iteratively denoising sampled noise from a prior distribution. The training stage of a diffusion model consists of a forward diffusion process and a reverse denoising process. Given an original data distribution $q(\mathbf{x}_0)$, the forward diffusion process gradually corrupts a data point $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ into a series of increasingly noisy data points $\mathbf{x}_{1:T} = \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T$ over T time steps. This process follows a Markov chain, where $q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$. Conversely, the reverse denoising process, denoted by $p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, aims to progressively reduce noise towards the original data distribution $q(\mathbf{x}_0)$ by predicting $\mathbf{x}_{t-1}$ from $\mathbf{x}_t$. The initial noise $\mathbf{x}_T$ is sampled from a predefined prior distribution $p(\mathbf{x}_T)$, and the denoising inference p_θ can be parametrized by a learnable neural network. Although the diffusion and denoising processes are agnostic to the data modality, the choice of prior distributions and Markov transition operators varies between continuous and discrete spaces.

In this work, we followed the settings of the discrete denoising diffusion proposed by Austin et al.[36] and Clement et al.[30]. In contrast with typical Gaussian diffusion models that operate in continuous state space, discrete denoising diffusion models introduce noise to categorical data using transition probability matrices in discrete state space. Let $x_t \in \{1, \cdots, K\}$ denote the categorical data with K categories and its one-hot encoding represented by $\mathbf{x}_t \in \mathbb{R}^K$. At time step t, the forward transition probabilities can be denoted by a matrix $\mathbf{Q}_t \in \mathbb{R}^{K\times K}$, where $[\mathbf{Q}_t]_{ij} = q(x_t = j|x_{t-1} = i)$ is the probability of transitioning from category i to category j. Therefore, the discrete transition kernel in the diffusion process is defined as

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{Cat}(\mathbf{x}_t; \mathbf{p} = \mathbf{x}_{t-1}\mathbf{Q}_t), \tag{1}$$

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathrm{Cat}(\mathbf{x}_t; \mathbf{p} = \mathbf{x}_0\overline{\mathbf{Q}}_t), \text{ with } \overline{\mathbf{Q}}_t = \mathbf{Q}_1\mathbf{Q}_2\cdots\mathbf{Q}_t, \tag{2}$$

where $\mathrm{Cat}(\mathbf{x}; \mathbf{p})$ represents a categorical distribution over $\mathbf{x}_t$ with probabilities determined by $\mathbf{p} \in \mathbb{R}^K$. As the diffusion process has a Markov chain, the transition matrix from $\mathbf{x}_0$ to $\mathbf{x}_t$ can be written as a closed form in equation (2) with $\overline{\mathbf{Q}}_t = \mathbf{Q}_1\mathbf{Q}_2\cdots\mathbf{Q}_t$. This property enables efficient sampling of $\mathbf{x}_t$ at arbitrary time steps without recursively applying noise. Following the Bayesian theorem, the calculation of posterior distribution (with the derivation in Supplementary Information Section 3) from time step t to $t-1$ can be written as

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \propto \mathbf{x}_t\mathbf{Q}_t^T \odot \mathbf{x}_0\overline{\mathbf{Q}}_{t-1}, \tag{3}$$

where $\odot$ is a Hadamard (element-wise) product. The posterior $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ is equivalent to $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ owing to its Markov property. Thus, the clean data $\mathbf{x}_0$ is introduced for denoising estimation and can be used as the target of the denoising neural network. In MapDiff, we introduce two simple but effective choices for the transition matrix $\mathbf{Q}_t$: uniform transition[36] and marginal transition[30]. The uniform transition is parametrized by $\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t\mathbf{1}_K\mathbf{1}_K^T/K$, where $K = 20$ represents the number of native amino acid types and the noise schedule $\beta_t \in [0, 1]$. Similarly, the marginal transition is parametrized by $\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t\mathbf{1}_K\mathbf{p}^T$, where $\mathbf{p} \in \mathbb{R}^{20}$ denotes the marginal probability distribution of AA types in the training data. All matrix values are strictly positive, and each row sums to one, ensuring the conservation of probability mass. Given these properties, along with the condition $\lim_{t\to T}\beta_t = 1$, $q(\mathbf{x}_t)$ can converge to a stationary uniform or marginal distribution, regardless of the initial $\mathbf{x}_0$.

Residue graph construction

IPF prediction aims to generate a feasible AA sequence that can fold into a desired backbone structure. Given a target protein of length L, we present it as a proximity residue graph $\mathcal{G} = (\mathbf{X}, \mathbf{A}, \mathbf{E})$, where each node denotes an AA residue within the protein. The node features $\mathbf{X} = [\mathbf{X}^{\mathrm{aa}}, \mathbf{X}^{\mathrm{pos}}, \mathbf{X}^{\mathrm{prop}}]$ encode the AA residue types, 3D spatial coordinates and geometric properties. The adjacency matrix $\mathbf{A} \in \{0,1\}^{N\times N}$ is constructed using the k-nearest-neighbour algorithm. Specifically, each node is connected to a maximum of k other nodes within a cutoff distance smaller than 30 Å. The edge feature matrix $\mathbf{E} \in \mathbb{R}^{M\times 93}$ illustrates the spatial and sequential relationships between the connected nodes. More details on the graph feature construction are provided in Supplementary Information Section 4. For sequence generation, we define a discrete denoising process on the types of noisy AA residues $\mathbf{X}_t^{\mathrm{aa}} \in \mathbb{R}^{N\times 20}$ at time t. Conditioned on the noise graph $\mathcal{G}_t$, this process is subject to iteratively refine noise $\mathbf{X}_t^{\mathrm{aa}}$ towards a clean $\mathbf{X}_0^{\mathrm{aa}} = \mathbf{X}^{\mathrm{aa}}$, which is predicted by our mask-prior-guided denoising network.

IPF denoising diffusion process

Discrete diffusion process. In the diffusion process, we incrementally introduced discrete noise to the clean AA residues over a number of time steps $t \in \{1, \cdots, T\}$, which resulted in transforming the original data distribution to a simple uniform or marginal distribution. Given a clean AA sequence $\mathbf{X}_0^{\mathrm{aa}} = \{\mathbf{x}_0^i \in \mathbb{R}^{1\times 20} | 1 \le i \le N\}$, we used a cumulative transition matrix $\overline{\mathbf{Q}}_t$ to independently add noise to each AA residue at arbitrary step t

$$q(\mathbf{x}_t^i|\mathbf{x}_0^i) = \mathrm{Cat}(\mathbf{x}_t^i; \mathbf{p} = \mathbf{x}_0^i\overline{\mathbf{Q}}_t), \text{with } \overline{\mathbf{Q}}_t = \mathbf{Q}_1\mathbf{Q}_2\cdots\mathbf{Q}_t, \tag{4}$$

$$q(\mathbf{X}_t^{\mathrm{aa}}|\mathbf{X}_0^{\mathrm{aa}}) = \prod_{1\le i\le N} q(\mathbf{x}_t^i|\mathbf{x}_0^i), \tag{5}$$

where $\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t \mathbf{1}_K \mathbf{1}_K^T/K$, and K denotes the number of native AA types (that is, $K = 20$). The weight of the noise, $\beta_t \in [0,1]$ was determined by a common cosine schedule[37].

Training objective of denoising network. The denoising neural network, denoted by ϕ_θ, is an essential component to reverse the noise process in diffusion models. In our framework, the network takes a noise residue graph $\mathcal{G}_t = (\mathbf{X}_t, \mathbf{A}, \mathbf{E})$ as input and aims to predict the clean AA residues $\mathbf{X}_0^{\mathrm{aa}}$. Specifically, we designed a mask-prior-guided denoising network ϕ_θ to effectively capture inherent structural information and learn the underlying data distribution. To train the learnable network ϕ_θ, the objective is to minimize the cross-entropy loss between the predicted AA probabilities and the real AA types over all nodes.

Reverse denoising process. After the denoising network has been trained, it can be used to generate new AA sequences through an iterative denoising process. In this study, we first used the denoising network ϕ_θ to estimate the generative distribution $\hat{p}_\theta(\hat{\mathbf{x}}_0^i|\mathbf{x}_t^i)$ for each AA residue. Then the reverse denoising distribution $p_\theta(\mathbf{x}_{t-1}^i|\mathbf{x}_t^i)$ was parametrized by combining the posterior distribution with the marginalized network predictions as follows:

$$p_\theta\left(\mathbf{x}_{t-1}^i|\mathbf{x}_t^i\right) \propto \sum_{\hat{\mathbf{x}}_0^i} q\left(\mathbf{x}_{t-1}^i|\mathbf{x}_t^i, \hat{\mathbf{x}}_0^i\right) \hat{p}_\theta\left(\hat{\mathbf{x}}_0^i|\mathbf{x}_t^i\right), \tag{6}$$

$$p_\theta\left(\mathbf{X}_{t-1}^{\mathrm{aa}}|\mathbf{X}_t^{\mathrm{aa}}\right) = \prod_{1\le i\le N} p_\theta\left(\mathbf{x}_{t-1}^i|\mathbf{x}_t^i\right), \tag{7}$$

where $\hat{\mathbf{x}}_0^i$ represents the predicted probability distribution for the ith residue $\mathbf{x}_0^i$. The posterior distribution is defined as

$$\begin{aligned} q\left(\mathbf{x}_{t-1}^i|\mathbf{x}_t^i, \hat{\mathbf{x}}_0^i\right) &= \frac{q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \hat{\mathbf{x}}_0^i) q(\mathbf{x}_{t-1}^i|\hat{\mathbf{x}}_0^i)}{q(\mathbf{x}_t^i|\hat{\mathbf{x}}_0^i)}, \\ &= \mathrm{Cat}\left(\mathbf{x}_{t-1}^i; \mathbf{p} = \frac{\mathbf{x}_t^i \mathbf{Q}_t^T \odot \hat{\mathbf{x}}_0^i \overline{\mathbf{Q}}_{t-1}}{\hat{\mathbf{x}}_0^i \overline{\mathbf{Q}}_t (\mathbf{x}_t^i)^T}\right). \end{aligned} \tag{8}$$

By applying the reverse denoising process, the generation of less-noisy $\mathbf{X}_{t-1}^{\mathrm{aa}}$ from $\mathbf{X}_t^{\mathrm{aa}}$ is feasible (derivation in Supplementary Information Section 3). The denoised result is determined by the predicted residues from the denoising neural network, as well as the predefined transition matrices at steps t and $t-1$. To generate a new AA sequence, the complete generative process begins with a random noise from the independent prior distribution $p(\mathbf{x}_T)$. The initial noise is then iteratively denoised at each time step using the reverse denoising process, gradually converging to a desired sequence conditioned on the given graph $\mathcal{G}$.

DDIM with Monte-Carlo dropout. Although discrete diffusion models have demonstrated impressive generation ability in many fields, the generative process suffers from two limitations that hinder their success in IPF prediction. First, the generative process is inherently computationally inefficient due to the numerous denoising steps involved, which require a sequential Markovian forward pass for the iterative generation. Second, the categorical distribution used for denoising sampling lacks sufficient uncertainty estimation. Many studies indicate that the logits produced by deep neural networks do not accurately represent the true probabilities. Typically, the predictions tend to be overconfident, leading to a discrepancy between the predicted probabilities and the actual distribution. As the generative process iteratively draws samples from the estimated categorical distribution, insufficient uncertainty estimation will accumulate sampling errors and result in unsatisfactory performance.

To accelerate the generative process and improve uncertainty estimation, we propose a discrete sampling method by combining DDIM with Monte-Carlo dropout. DDIM[21] is a widely used method that improves the generation efficiency of diffusion models in continuous space. It defines the generative process as the reverse of a deterministic and non-Markovian diffusion process, making it possible to skip certain denoising steps during generation. As discrete diffusion models possess analogous properties, Yi et al. (2023)[38] extended DDIM into discrete space for IPF prediction. Similarly, we define the discrete DDIM sampling to the posterior distribution by

$$q\left(\mathbf{x}_{t-k}^i|\mathbf{x}_t^i, \hat{\mathbf{x}}_0^i\right) = \mathrm{Cat}\left(\mathbf{x}_{t-k}^i; \mathbf{p} = \frac{\mathbf{x}_t^i \mathbf{Q}_t^T \cdots \mathbf{Q}_{t-k}^T \odot \hat{\mathbf{x}}_0^i \overline{\mathbf{Q}}_{t-k}}{\hat{\mathbf{x}}_0^i \overline{\mathbf{Q}}_t (\mathbf{x}_t^i)^T}\right), \tag{9}$$

where k is the number of skipping steps.

Then we introduce the application of Monte-Carlo dropout within the generative process, a technique designed to enhance prediction uncertainty in neural networks. Specifically, we use dropout not only to prevent overfitting during the training of our denoising network, but also to maintain its activation in the inference stage. By keeping dropout enabled and running multiple forward passes (Monte-Carlo samples) during inference, we generate a prediction distribution for each input, as opposed to a single-point estimation. To improve uncertainty estimation, we aggregate the predictions by taking a mean pooling over all output logits corresponding to the same input. This operation leads to the predicted logits that perform reduced estimation bias, and their normalized probabilities can more accurately reflect the actual distribution. Therefore, we can leverage Monte-Carlo dropout to enhance the generative process towards more reliable samplings.

Mask-prior-guided denoising network

In diffusion model applications, the denoising network plays a crucial role in generation performance. We have developed a mask-prior-guided denoising network, integrating both structural information and residue interactions for enhanced protein sequence prediction. Our denoising network architecture encompasses a structure-based sequence predictor, a pretrained mask sequence designer and a mask ratio adaptor.

Structure-based sequence predictor. We adopt an EGNN with a global-aware module as the structure-based sequence predictor, which generates a full AA sequence from the backbone structure. EGNN is a type of graph neural network that satisfies equivariance operations for the special Euclidean group SE(3). It preserves geometric and spatial relationships of 3D coordinates within the message-passing framework. Given a noise residue graph, we use $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_N]$ to denote the initial node embeddings, which are derived from the noisy AA types and geometric properties. The coordinates of each node are represented by $\mathbf{X}^{\mathrm{pos}} = [\mathbf{x}_1^{\mathrm{pos}}, \mathbf{x}_2^{\mathrm{pos}}, \cdots \mathbf{x}_N^{\mathrm{pos}}]$, whereas the edge features are denoted by $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \cdots \mathbf{e}_M]$. In this setting, EGNN consists of a stack of equivariant graph convolutional layers (EGCL) for the node and edge information propagation, which are defined as

$$\mathbf{e}_{ij}^{(l+1)} = \phi_e\left(\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)}, \| \mathbf{x}_i^{(l)} - \mathbf{x}_j^{(l)} \|^2, \mathbf{e}_{ij}^{(l)}\right), \tag{10}$$

$$\hat{\mathbf{h}}_i^{(l+1)} = \phi_h\left(\mathbf{h}_i^{(l)}, \sum_{j\in\mathcal{N}(i)} w_{ij} \mathbf{e}_{ij}^{(l+1)}\right), \tag{11}$$

$$\mathbf{x}_i^{(l+1)} = \mathbf{x}_i^{(l)} + \frac{1}{N_i} \sum_{j\in\mathcal{N}(i)} \left(\mathbf{x}_i^{(l)} - \mathbf{x}_j^{(l)}\right) \phi_x\left(\mathbf{e}_{ij}^{(l+1)}\right), \tag{12}$$

where l denotes the lth EGCL layer, $\mathbf{x}_i^{(0)} = \mathbf{x}_i^{\mathrm{pos}}$ and $w_{ij} = \mathrm{sigmoid}(\phi_w(\mathbf{e}_{ij}^{(l+1)}))$ is a soft estimated weight assigned to the specific edge

representation. All components ($\phi_e, \phi_h, \phi_x, \phi_w$) are learnable and parametrized by fully connected neural networks. In the information propagation, EGNN achieves equivariance to translations and rotations on the node coordinates $\mathbf{X}^{\text{pos}}$, and preserves invariant to group transformations on the node features $\mathbf{H}$ and edge features $\mathbf{E}$.

However, the vanilla EGNN only considers local neighbour aggregation while neglecting the global context. Some recent studies[13,39] have demonstrated the importance of global information in protein design. Therefore, we introduce a global-aware module in the EGCL layer, which incorporates the global pooling vector into the update of node representations: that is,

$$\mathbf{m}^{(l+1)} = \text{MeanPool}\left(\left\{\hat{\mathbf{h}}_i^{(l+1)}\right\}_{i\in\mathcal{G}}\right), \tag{13}$$

$$\mathbf{h}_i^{(l+1)} = \hat{\mathbf{h}}_i^{(l+1)} \odot \text{sigmoid}\left(\phi_m\left(\mathbf{m}^{(l+1)}, \hat{\mathbf{h}}_i^{(l+1)}\right)\right), \tag{14}$$

where MeanPool$(\cdot)$ is the mean pooling operation over all nodes within a residue graph. The global-aware module effectively integrates global context into modelling and only increases a linear computational cost. To predict the probabilities of residue types, the node representations from the last EGCL layer are fed into a fully connected classification layer with softmax function, which is defined as

$$\mathbf{p}_i^{\text{b}} = \text{softmax}\left(\mathbf{l}_i^{\text{b}}\right), \quad \mathbf{l}_i^{\text{b}} = \mathbf{h}_i^{(L)}\mathbf{W}_{\text{o}} + \mathbf{b}_{\text{o}}, \tag{15}$$

where $\mathbf{W}_{\text{o}} \in \mathbb{R}^{D_h \times 20}$ and $\mathbf{b}_{\text{o}} \in \mathbb{R}^{1\times 20}$ are a learnable weight matrix and a bias vector respectively.

Low-confidence residue selection and mask ratio adaptor. As previously mentioned, structural information alone can sometimes be insufficient to determine all residue identities. Certain flexible regions display a weaker correlation with the backbone structure but are strongly influenced by their sequential context. To enhance the denoising network's performance, we introduce a masked sequence designer module. This module refines the residues identified with low confidence in the base sequence predictor. We adopt an entropy-based residue selection strategy, as proposed by Zhou et al. (2023)[24], to identify these low-confidence residues. The entropy for the ith residue of the probability distribution $\mathbf{p}_i^b$ is calculated as

$$\text{ent}_i^{\text{b}} = -\sum_j \text{p}_{ij}^{\text{b}} \log\left(\text{p}_{ij}^{\text{b}}\right). \tag{16}$$

Given that entropy quantifies the uncertainty in a probability distribution, it can be used to locate the low-confidence predicted residues. Consequently, residues with the most entropy are masked, whereas the rest remain in a sequential context. The masked sequence designer aims to reconstruct the entire sequence by using the masked partial sequence in combination with the backbone structure. In addition, to account for the varying noise levels of the input sequence in diffusion models, we designed a simple mask ratio adaptor to dynamically determine the entropy mask percentage at different denoising steps: that is,

$$\text{mr}_t = \sin\left(\frac{\pi}{2}\beta_t\sigma\right) + m, \tag{17}$$

where $\beta_t \in [0,1]$ represents the noise weight at step t derived from the noise schedule, and σ and m are the predefined deviation and minimum mask ratio, respectively. With the increase of β_t, the mask ratio is proportional to its time step.

Mask-prior pretraining. To incorporate prior knowledge of sequential context, we pretrained the masked sequence designer by applying the masked language modelling objective proposed in BERT[40]. It is important to clarify that we used the same training data in the diffusion models for pretraining purposes, to avoid any information leakage from external sources. In this process, we randomly sampled a proportion of residues in the native AA sequences and replaced them with the masking procedures: (1) masking 80% of the selected residues using a special MASK type; (2) replacing 10% of the selected residues with other random residue types; and (3) keeping the remaining 10% residues unchanged. Subsequently, we input the partially masked sequences, along with structural information, into the masked sequence designer. The objective of the pretraining stage was to predict the original residue types from the masked residue representations using a cross-entropy loss function.

Masked sequence designer. We used an IPA network as the masked sequence designer. IPA is a geometry-aware attention mechanism designed to facilitate the fusion of residue representations and spatial relationships, enhancing the structure generation within AlphaFold2[15]. In this study, we repurposed the IPA module to refine low-confidence residues in the base sequence predictor. Given a mask AA sequence, we denote its residue representation as $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_N]$, which is derived from the residue types and positional encoding. To incorporate geometric information, as with the IPA implementation in Frame2seq[41], we constructed a pairwise distance representation $\mathbf{Z} = \{\mathbf{z}_{ij} \in \mathbb{R}^{1\times d_z} | 1 \le i \le N, 1 \le j \le N\}$ and rigid coordinate frames $\mathcal{T} = \{T_i := (\mathbf{R}_i \in \mathbb{R}^{3\times 3}, \mathbf{t}_i \in \mathbb{R}^3) | 1 \le i \le N\}$. The pairwise representation $\mathbf{Z}$ was obtained by calculating interresidue spatial distances and relative sequence positions. The rigid coordinate frames were constructed from the coordinates of backbone atoms using a Gram–Schmidt process, providing a consistent local reference for ensuring the invariance of IPA to global Euclidean transformations. Subsequently, we took the residue representation, pairwise distance representation and rigid coordinate frames as inputs, and fed them into a stack of IPA layers for representation learning, which is defined as

$$\mathbf{S}^{(l+1)}, \mathbf{Z}^{(l+1)} = \text{IPA}(\mathbf{S}^{(l)}, \mathbf{Z}^{(l)}, \mathcal{T}). \tag{18}$$

The IPA network follows the self-attention mechanism. However, it enhances the general attention queries, keys and values by incorporating 3D points that are generated in the rigid coordinate frame of each residue. This operation ensures that the updated residue and pair representations remain invariant by global rotations and translations. More details on the IPA feature construction and algorithm implementation are provided in Supplementary Information Section 6. For the ith residue, the predicted probability distribution and entropy in the masked sequence designer are calculated as

$$\mathbf{p}_i^{\text{m}} = \text{softmax}(\mathbf{l}_i^{\text{m}}), \quad \mathbf{l}_i^{\text{m}} = \mathbf{h}_i^{(L)}\mathbf{W}_{\text{m}} + \mathbf{b}_{\text{m}}, \tag{19}$$

$$\text{ent}_i^{\text{m}} = -\sum_j \text{p}_{ij}^{\text{m}} \log(\text{p}_{ij}^{\text{m}}), \tag{20}$$

where $\mathbf{W}_{\text{m}} \in \mathbb{R}^{D_s\times 20}$ and $\mathbf{b}_{\text{m}} \in \mathbb{R}^{1\times 20}$ are the learnable weight matrix and bias vector, respectively. The training objective was to jointly minimize the cross-entropy losses for both the base sequence predictor and masked sequence designer. In the inference stage, we calculated the final predicted probability by weighting the output logits based on their entropy as

$$\mathbf{l}_i^{\text{f}} = \frac{\exp\left(-\text{ent}_i^{\text{b}}\right)}{\exp\left(-\text{ent}_i^{\text{b}}\right) + \exp\left(-\text{ent}_i^{\text{m}}\right)}\mathbf{l}_i^{\text{b}} + \frac{\exp\left(-\text{ent}_i^{\text{m}}\right)}{\exp\left(-\text{ent}_i^{\text{b}}\right) + \exp\left(-\text{ent}_i^{\text{m}}\right)}\mathbf{l}_i^{\text{m}}. \tag{21}$$

$$\mathbf{p}_i^{\text{f}} = \text{softmax}\left(\mathbf{l}_i^{\text{f}}\right). \tag{22}$$

By incorporating the mask-prior denoising network into the discrete denoising diffusion process, our framework enhanced the denoising trajectories, leading to more accurate predictions of protein sequences.

Experimental setting

Primary datasets. We evaluated MapDiff on experimentally validated protein structures curated from well-established databases. The CATH database[25] is widely used in inverse folding research, enabling fair comparisons across different methodologies. It classifies proteins into hierarchical levels based on class, architecture, topology and homologous superfamily, with filtering to reduce redundancy and ensure structural diversity. Following previous studies[13,26,27], proteins are partitioned based on their CATH topology classification codes, ensuring that the training, validation and test sets contain non-overlapping topologies. This partitioning strategy provided a robust evaluation of the model's generalization to unseen proteins. For CATH 4.2, the dataset consisted of 18,024 structures for training, 608 for validation and 1,120 for testing. Similarly, in CATH 4.3, we followed the topology classification approach in ESM-IF[27], resulting in 16,630 proteins for training, 1,516 for validation and 1,864 for testing. By including both CATH 4.2 and CATH 4.3, we assessed the stability of model performance across dataset versions, ensuring robustness to updates in protein-structure databases.

Zero-shot generalization datasets. To further assess MapDiff's zero-shot generalization ability, we evaluated it on the two independent TS50 and PDB2022 datasets. TS50 (ref. 5) is a commonly used benchmark for protein-sequence design, consisting of 50 diverse protein chains covering different structural classes. PDB2022 includes single-chain structures published in the Protein Data Bank (PDB)[42] between 5 January 2022 and 26 October 2022, curated by Zhou et al.[24], with protein length ≤500 and resolution ≤2.5 Å. This dataset consists of 1,975 proteins published after those in the CATH dataset, ensuring a strict time-based test 'split' to evaluate real-world temporal generalization. Both datasets are entirely separate from the CATH-derived training set, minimizing data leakage and providing a robust evaluation of structural and temporal generalization.

Baselines. We compared MapDiff with recent deep-graph models for inverse protein folding, including StructGNN[26], GraphTrans[26], GVP[43], AlphaDesign[44], ProteinMPNN[1], PiFold[13], LM-Design[45] and GRADE-IF[1]. To ensure a reliable and fair comparison, we reproduced the open-source and four most state-of-the-art baselines (ProteinMPNN, PiFold, LM-Design and GRADE-IF) under identical settings in our experiments. ProteinMPNN uses a message-passing neural network to encode structure features, and a random decoding scheme to generate protein sequences. PiFold introduces a residue featurizer to extract distance, angle and direction features. It proposes a PiGNN encoder to learn expressive residue representations, enabling the generation of protein sequences in a one-shot manner. LM-Design uses structure-based models as encoders and incorporates the protein language model ESM as a protein designer to refine the generated sequences. GRADE-IF employs EGNN to learn residue representations from protein structures, and it adopts the graph denoising diffusion model to iteratively generate feasible sequences. All baselines were implemented following the default hyperparameter settings in their original papers.

Implementation set-up. MapDiff is implemented in Python v.3.8 and PyTorch v.1.13.1 (ref. 46), along with functions from BioPython v.1.81 (ref. 47), PyG v.2.4.0 (ref. 48), Scikit-learn v.1.0.2 (ref. 49), NumPy v.1.22.3 (ref. 50) and RDKit v.2023.3.3 (ref. 51). It consists of two training stages: mask-prior pretraining and denoising diffusion model training, both of which use the same CATH 4.2/4.3 training set. The batch size was set to eight, and the models were trained up to 200 epochs in pretraining and 100 epochs in denoising training. We employed the Adam optimizer with a one-cycle scheduler for parameter optimization, setting the peak learning rate to 5×10^{-4}. In the denoising network, the structure-based sequence predictor consisted of six global-aware EGCL layers, each with 128 hidden dimensions. In addition, the masked sequence designer stacked six layers of IPA, each with 128 hidden dimensions and four attention heads. The dropout rate was set to 0.2 in both the EGCL and IPA layers. A cosine schedule was applied to control the noise weight at each time step, with a total of 500 time steps. During sampling inference, the skip steps for DDIM were configured to 100, and the Monte-Carlo forward passes were set to 50. For the mask ratio adaptor, we set the minimum mask ratio to 0.4 and the deviation to 0.2. All experiments were conducted on a single Tesla A100 GPU. Following the regular evaluation in deep learning, the best-performing model was selected based on the epoch that provided the highest recovery on the validation set. After that, this selected model was subsequently used to evaluate performance on the test set. For the foldability analysis, we applied a single AlphaFold2 pTM model (that is, model_1_ptm) with three recycles to balance accuracy and computational efficiency. Multiple sequence alignment information was generated for each sequence using the MMSeqs2 (refs. 52,53) server provided by ColabFold[54]. We provide the algorithm details for the training and sampling inference in Supplementary Information Section 5, and the scalability study in Supplementary Information Section 8 and Supplementary Fig. 4.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

The experimental data used in this work are available at https://github.com/peizhenbai/MapDiff/tree/main/data. All data were publicly collected from the following resources. The CATH 4.2 dataset can be found at https://github.com/dauparas/ProteinMPNN; the CATH 4.3 dataset can be found at https://github.com/BytedProtein/ByProt; the PDB2022 dataset can be found at https://github.com/veghen/ProRefiner and the TS50 dataset can be found at https://github.com/A4Bio/PiFold. The protein-structure data were obtained from Protein Data Bank at https://www.rcsb.org/ with the corresponding PDB IDs. Source data are provided with this paper.

Code availability

The source code and implementation details of MapDiff are available via GitHub at https://github.com/peizhenbai/MapDiff and via CodeOcean at https://doi.org/10.24433/CO.3441652.v1 (ref. 55). The code is also available via Zenodo at https://doi.org/10.5281/zenodo.15162932 (ref. 56).

References

1. Dauparas, J. et al. Robust deep learning-based protein sequence design using ProteinMPNN. *Science* **378**, 49–56 (2022).
2. Høie, M. H. et al. Antifold: improved antibody structure-based design using inverse folding. *Bioinform. Adv.* **5**, vbae202 (2025).
3. Alford, R. F. et al. The rosetta all-atom energy function for macromolecular modeling and design. *J. Chem. Theory Comput.* **13**, 3031–3048 (2017).
4. Wu, Z., Johnston, K. E., Arnold, F. H. & Yang, K. K. Protein sequence design with deep generative models. *Curr. Opin. Chem. Biol.* **65**, 18–27 (2021).
5. Li, Z., Yang, Y., Faraggi, E., Zhan, J. & Zhou, Y. Direct prediction of profiles of sequences compatible with a protein structure by neural networks with fragment-based local and energy-based nonlocal profiles. *Proteins* **82**, 2565–2573 (2014).
6. O'Connell, J. et al. Spin2: predicting sequence profiles from protein structures using deep neural networks. *Proteins* **86**, 629–633 (2018).

7. Anand, N. et al. Protein sequence design with a learned potential. *Nat. Commun.* **13**, 746 (2022).
8. Towse, C.-L. & Daggett, V. When a domain is not a domain, and why it is important to properly filter proteins in databases: conflicting definitions and fold classification systems for structural domains make filtering of such databases imperative. *Bioessays* **34**, 1060–1069 (2012).
9. Li, B., Tian, J., Zhang, Z., Feng, H. & Li, X. Multitask non-autoregressive model for human motion prediction. *IEEE Trans. Image Process.* **30**, 2562–2574 (2020).
10. Martínez-González, A., Villamizar, M. & Odobez, J.-M. Pose transformers (POTR): human motion prediction with non-autoregressive transformers. In *Proc. IEEE/CVF International Conference on Computer Vision* (eds Sharp, A. et al.) 2276–2284 (IEEE, 2021).
11. Starr, T. N. & Thornton, J. W. Epistasis in protein evolution. *Protein Sci.* **25**, 1204–1218 (2016).
12. Xu, Y. et al. Anytime sampling for autoregressive models via ordered autoencoding. In *Proc. International Conference on Learning Representations* (eds Oh, A. et al.) 1024 (ICLR, 2021).
13. Gao, Z., Tan, C. & Li, S. Z. Pifold: toward effective and efficient protein inverse folding. In *Proc. International Conference on Learning Representations* (eds Nickel, M. et al.) 3370 (ICLR, 2023).
14. Lyu, S., Sowlati-Hashjin, S. & Garton, M. Variational autoencoder for design of synthetic viral vector serotypes. *Nat. Mach. Intell.* **6**, 1–14 (2024).
15. Jumper, J. et al. Highly accurate protein structure prediction with alphafold. *Nature* **596**, 583–589 (2021).
16. Abramson, J. et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature* **630**, 493–500 (2024).
17. Watson, J. L. et al. De novo design of protein structure and function with RFdiffusion. *Nature* **620**, 1089–1100 (2023).
18. Baek, M. et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science* **373**, 871–876 (2021).
19. Jing, B., Corso, G., Chang, J., Barzilay, R. & Jaakkola, T. Torsional diffusion for molecular conformer generation. *Adv. Neural Inf. Process. Syst.* **35**, 24240–24253 (2022).
20. Schneuing, A. et al. Structure-based drug design with equivariant diffusion models. *Nat. Comput. Sci.* **4**, 899–909 (2024).
21. Song, J., Meng, C. & Ermon, S. Denoising diffusion implicit models. In *Proc. International Conference on Learning Representations* (eds Oh, A. et al.) 1080 (ICLR, 2021).
22. Gal, Y. & Ghahramani, Z. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In *Proc. International Conference on Machine Learning* (eds Balcan, M. F. et al.) 1050–1059 (PMLR, 2016).
23. Satorras, V. G., Hoogeboom, E. & Welling, M. E(n) equivariant graph neural networks. In *Proc. International Conference on Machine Learning* (eds Meila, M. et al.) 9323–9332 (PMLR, 2021).
24. Zhou, X. et al. Prorefiner: an entropy-based refining strategy for inverse protein folding with global graph attention. *Nat. Commun.* **14**, 7434 (2023).
25. Orengo, C. A. et al. Cath–a hierarchic classification of protein domain structures. *Structure* **5**, 1093–1109 (1997).
26. Ingraham, J., Garg, V., Barzilay, R. & Jaakkola, T. Generative models for graph-based protein design. *Adv. Neural Inf. Process. Syst.* **32**, 15820–15831 (2019).
27. Hsu, C. et al. Learning inverse folding from millions of predicted structures. In *Proc. International Conference on Machine Learning* (eds Chaudhuri, K. et al.) 8946–8970 (PMLR, 2022).
28. Löffler, P., Schmitz, S., Hupfeld, E., Sterner, R. & Merkl, R. Rosetta: MSF: a modular framework for multi-state computational protein design. *PLoS Comput. Biol.* **13**, e1005600 (2017).
29. Henikoff, S. & Henikoff, J. G. Amino acid substitution matrices from protein blocks. *Proc. Natl Acad. Sci. USA* **89**, 10915–10919 (1992).
30. Vignac, C. et al. Digress: discrete denoising diffusion for graph generation. In *Proc. International Conference on Learning Representations* (eds Nickel, M. et al.) 2829 (ICLR, 2023).
31. Limpert, E., Stahel, W. A. & Abbt, M. Log–normal distributions across the sciences: keys and clues: on the charms of statistics, and how mechanical models resembling gambling machines offer a link to a handy way to characterize log–normal distributions, which can provide deeper insight into variability and probability—normal or log–normal: that is the question. *BioScience* **51**, 341–352 (2001).
32. Bloch, V. et al. The H-NS dimerization domain defines a new fold contributing to DNA recognition. *Nat. Struct. Mol. Biol.* **10**, 212–218 (2003).
33. Huang, Y.-C. et al. The flexible and clustered lysine residues of human ribonuclease 7 are critical for membrane permeability and antimicrobial activity. *J. Biol. Chem.* **282**, 4626–4633 (2007).
34. Mansy, S. S. et al. Structure and evolutionary analysis of a non-biological atp-binding protein. *J. Mol. Biol.* **371**, 501–513 (2007).
35. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. Basic local alignment search tool. *J. Mol. Biol.* **215**, 403–410 (1990).
36. Austin, J., Johnson, D. D., Ho, J., Tarlow, D. & Van Den Berg, R. Structured denoising diffusion models in discrete state-spaces. *Adv. Neural Inf. Process. Syst.* **34**, 17981–17993 (2021).
37. Nichol, A. Q. & Dhariwal, P. Improved denoising diffusion probabilistic models. In *Proc. International Conference on Machine Learning* (eds Meila, M. et al.) 8162–8171 (PMLR, 2021).
38. Yi, K., Zhou, B., Shen, Y., Liò, P. & Wang, Y. Graph denoising diffusion for inverse protein folding. *Adv. in Neural Inf. Process. Syst.* **36**, 10238–10257 (2023).
39. Tan, C., Gao, Z., Xia, J., Hu, B. & Li, S. Z. Global-context aware generative protein design. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (eds Narayanan, S. et al.) 1–5 (IEEE, 2023).
40. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol. 1 (eds Burstein, J. et al.) 4171–4186 (ACL, 2019).
41. Akpinaroglu, D. et al. Structure-conditioned masked language models for protein sequence design generalize beyond the native sequence space. Preprint at *bioRxiv* https://doi.org/10.1101/2023.12.15.571823 (2023).
42. Berman, H. M. et al. The protein data bank. *Nucleic Acids Res.* **28**, 235–242 (2000).
43. Jing, B., Eismann, S., Suriana, P., Townshend, R. J. L. & Dror, R. Learning from protein structure with geometric vector perceptrons. In *Proc. International Conference on Learning Representations* (eds Oh, A. et al.) 1954 (ICLR, 2021).
44. Gao, Z., Tan, C. & Li, S. Z. Alphadesign: a graph protein design method and benchmark on alphafolddb. Preprint at https://arxiv.org/abs/2202.01079 (2022).
45. Zheng, Z. et al. Structure-informed language models are protein designers. In *Proc. International Conference on Machine Learning* (eds Krause, A. et al.) 42317–42338 (PMLR, 2023).
46. Paszke, A. et al. Automatic differentiation in PyTorch. In *Proc. NIPS Workshop on Autodiff* (eds Wiltschko, A. et al.) 8 (NIPS, 2017).
47. Cock, P. J. et al. BioPython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25**, 1422–1423 (2009).
48. Fey, M. & Lenssen, J. E. Fast graph representation learning with PyTorch Geometric. In *Proc. ICLR Workshop on Representation Learning on Graphs and Manifolds* (eds Battaglia, P. et al.) (ICLR, 2019).

49. Pedregosa, F. et al. Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
50. Harris, C. R. et al. Array programming with NumPy. *Nature* **585**, 357–362 (2020).
51. Landrum, G. RDKit: open-source cheminformatics. www.rdkit.org (2006).
52. Steinegger, M. & Söding, J. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat. Biotechnol.* **35**, 1026–1028 (2017).
53. Mirdita, M., Steinegger, M. & Söding, J. MMseqs2 desktop and local web server app for fast, interactive sequence searches. *Bioinformatics* **35**, 2856–2858 (2019).
54. Mirdita, M. et al. Colabfold: making protein folding accessible to all. *Nat. Methods* **19**, 679–682 (2022).
55. Bai, P. et al. Mask prior-guided denoising diffusion improves inverse protein folding. *Code Ocean* https://doi.org/10.24433/CO.3441652.v1 (2025).
56. Bai, P. et al. peizhenbai/MapDiff: v.1.0.0. *Zenodo* https://doi.org/10.5281/zenodo.15162932 (2025).
57. Kabsch, W. & Sander, C. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers* **22**, 2577–2637 (1983).
58. *Schrödinger Release 2023–4: Maestro* (Schrödinger, 2023).

Acknowledgements

We are grateful to T. Ucar, X. Song and S. Zhou for their invaluable suggestions on the work. P.B. received the Faculty of Engineering Research Scholarship at the University of Sheffield.

Author contributions

P.B. developed the models and conceived and designed the experiments under the guidance of L.D.M., R.C.W., O.R. and H.L. F.M., X.L. and P.B. contributed to the analysis tools, performed the experiments and conducted method comparisons. All authors contributed to analysing the data and writing the paper.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-025-01042-6.

Correspondence and requests for materials should be addressed to Haiping Lu.

Peer review information *Nature Machine Intelligence* thanks Rohith Krishna and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

nature portfolio

Corresponding author(s): Haiping Lu

Last updated by author(s): Apr 9, 2025

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☐	☒	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	Python 3.8, BioPython 1.81
Data analysis	Python 3.8, BioPython 1.81, RDKit 2023.3.3, PyTorch 1.13.1, PyG 2.4.0, Scikit-learn 1.0.2, NumPy 1.22.3 and Maestro Schrödinger 2023.4. We also make the source code of this study available at GitHub (https://github.com/peizhenbai/MapDiff) and Code Ocean (https://doi.org/10.24433/CO.3441652.v1).

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our policy

Our manuscript includes a data availability statement. All experimental data used in this work are publicly available and collected from the following resources. The

CATH 4.2 dataset can be found at https://github.com/dauparas/ProteinMPNN; the CATH 4.3 dataset can be found at https://github.com/BytedProtein/ByProt; the PDB2022 dataset can be found at https://github.com/veghen/ProRefiner and the TS50 dataset can be found at https://github.com/A4Bio/PiFold. The protein structure data is obtained from Protein Data Bank https://www.rcsb.org with the corresponding PDB IDs.

Human research participants

Policy information about studies involving human research participants and Sex and Gender in Research.

Reporting on sex and gender	Not involve sex and gender.
Population characteristics	No population characteristics.
Recruitment	No participants were recruited.
Ethics oversight	*Identify the organization(s) that approved the study protocol.*

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	We studied four public datasets CATH 4.2, CATH 4.3, TS50 and PDB2022 with sample sizes 19k, 20k, 50 and 1.9k, respectively. The sample sizes were chosen based on three key considerations: i) to align with the scale of datasets commonly used in state-of-the-art works in this area; ii) to select highly reputable, publicly available datasets that are widely recognized; iii) to use experimentally validated structural data from the PDB database. In this way, our chosen sample sizes ensure a robust and fair performance evaluation against existing state-of-the-art methods.
Data exclusions	After the datasets were chosen as described above, no data was excluded in this work.
Replication	To verify the reproducibility of our experimental findings, we provided the source code, experimental data and trained weights at our public GitHub repository and in the CodeOcean capsule.
Randomization	Our strategies were based on random sample allocation. For the CATH dataset, we allocated data samples into experimental groups (splits) with topology-based data split, where the proteins are randomly assigned to training, validation and test sets based on their topology classification codes. As a result, there is no overlap of protein topology codes among the different sets. Moreover, we use another two distinct datasets, TS50 and PDB2022 to evaluate the zero-shot generalization of models. These two datasets have no overlap with the training data.
Blinding	We were blinded to the group allocation during data collection and analysis. The group allocation process was performed by computer script without any manual intervention.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☒	☐ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging