Accelerated Diffusion Models for Protein Structure Generation

Haowei Lin[*]
Peking University
linhaowei@pku.edu.cn

Shenshen Li[*]
Peking University
lssastar@pku.edu.cn

Yuzhe Wang[*]
Peking University
wangyuzhe_ccme@pku.edu.cn

Abstract

Proteins are workhorse molecules of life, and generation of novel protein structures from scratch that enable targeted functions, a task known as *de novo* protein design, has long been a challenging task in the field of bioengineering. Diffusion models have achieved considerable success on protein structure generation tasks, but these models generally suffer from slow sampling speed that limits their practicability. In this work, we present an accelerated diffusion-based sequence-conditioned generative model that is capable of generating high-quality protein structures with significantly improved efficency. We introduce pre-trained AlphaFold2 Evoformer module as feature extractor and implement the diffusion module based on AlphaFold2 Struture module. We apply several representative acceleration methods, namely DDIM, DPM-Solver and DPM-Solver++, to our proposed model and achieve remarkable speedup performance on the task of protein structure generation. We demonstrate that our simple yet effective model architecture combined with the speedup algorithms may further enhance the capability of diffusion models on *de novo* protein design tasks. The code is attached as supplemental material.

1 Introduction

Proteins are large macromolecules that carry out most of the biological functions necessary for life. They serve as antibodies that identify and neutralize pathogenic bacteria and viruses, enzymes that catalyse biochemical reactions as well as messengers that relay signals across neurons [29, 45]. A longstanding task in the field of bioengineering is designing proteins from scratch that enjoy specific biochemical properties which enable targeted functions, which is known as *de novo* protein design. The process of *de novo* protein design requires generation of both the backbone structure and the amino acid sequence, which is generally achieved via iterative refinement procedures.

In the past few decades, significant progress has been made towards computational protein design, and a wide range of novel proteins are successfully designed with atomic-level accuracy [11]. Recent success of AlphaFold2 (AF2) has also enhanced our understanding of the relationship between protein structure and its functions [14]. However, while existing approaches such as Rosetta [7, 20] and Learned Potential [3] have come to perform well for the fixed-backbone sequence generation problem, the task of protein structure generation remains a challenge. Most proposed methods use stochastic search algorithms based on handcrafted energy functions and heuristic sampling approaches to generate novel protein structures [16, 19, 1], and the performance of these methods are generally unsatisfactory in terms of robustness and generation quality.

[*] The author contribution is presented at the end of this report.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Diffusion probabilistic models (DPMs) [36, 10, 38] are emerging powerful generative models that have shown promising performance on a variety of tasks such as image generation [32] and 3D object generation [30]. Recently, diffusion models have been applied to the task of protein structure generation and have revolutionized the field of *de novo* protein design with state-of-the-art performance achieved [3, 27, 42, 40, 12, 43]. However, these diffusion-based generative models suffer from their slow sampling as they generally need a couple of or even tens of minutes to produce a valid protein structure, and slow inference procedure has severely limited the practicability of these models.

In this work, we propose a novel accelerated diffusion-based sequence-conditioned generative model that generates high-quality protein structures with significantly improved efficiency. Inspired by the model architecture of AlphaFold2 [14], we introduce pre-trained AF2 Evoformer module to extract features from a given amino acid sequence and hence guide the generation process. The diffusion module is based on AF2 Structure module and is fully trainable following the scheme of general diffusion models. To handle the existing problem of slow sampling, we test a variety of accelerated sampling algorithms including DDIM [37], DPM-Solver [24]and DPM-Solver++ [25] on our proposed diffusion model and achieve remarkable speedup performance. We demonstrate that the model we introduce is capable of generating high-quality protein structures with significant efficiency under the guidance of the sequence information with a simple but effective architecture. To the best of our knowledge, this is the first work that combines diffusion-based generative models with accelerated sampling methods in the field of protein structure generation. We believe that our proposed method can provide convenience for further model refinement, as well as improving the efficiency of antibody design and drug discovery.

2 Related Work

2.1 Diffusion-based protein structure generation

A variety of generative deep learning architectures have been applied to the task of protein structure generation before the emergence of diffusion models, including generative adversarial network (GAN)-based [2], long short-term memory (LSTM) GAN-based [33] and variational autoencoder (VAE)-based [8] models. Several works have implemented diffusion models to protein structure generation task and made notable progress. Trippe et al. [40] adopt an E(3)-equivariant graph neural network (EGNN) to model the coordinates of protein residues and perform diffusion on the 3D Cartesian coordinates of the residues. Anand & Achim [3] train an equivariant transformer with invariant point attention (IPA) inspired by AF2, which simultaneously generates the 3D coordinates of Cα atoms, the amino acid sequence and the internal angles defining the orientations of side chains. Luo et al. [27] perform diffusion for generating antibody fragments' structure and sequence by modeling 3D coordinates using an equivariant neural network. ProteinSGM proposed by Lee & Kim [21] implement a score-based generative model that generates pairwise angles and distances between amino acids and recover the protein structure via Rosetta folding algorithm [44]. Wu et al. [43] propose a model that perform diffusion on a minimal set of internal angles required to specify a protein backbone.

Recent works of Chroma [12] proposed by Ingraham et al. and RFDiffusion [42] proposed by Watson et al. have achieved remarkable success in the task of *de novo* protein design. Ingraham et al. implement an efficient neural architecture for protein systems based on random graph neural networks that enables long-range reasoning with sub-quadratic scaling, equivariant layers for efficiently synthesizing 3D structures of proteins and a general low-temperature sampling algorithm for diffusion models. They demonstrate that Chroma effectively realizes *de novo* protein design as Bayesian inference under external constraints including symmetries, substructure, shape, semantics, and even natural language prompts. Watson et al. fine-tune the RoseTTAFold structure prediction network [4] on protein structure denoising tasks and obtain a generative model of protein backbones that achieve outstanding performance on a series of protein design tasks. Success of Chroma and RFDiffusion has proven the power and potential of diffusion models in the field of protein design.

2.2 Accelerated sampling for diffusion models

Slow sampling speed of DPMs has long been a critical bottleneck for their adoption in downstream tasks, and hence a variety of fast samplers for DPMs are carried out to handle such inefficiency. Existing fast samplers for DPMs can be devided into two categories. The first category is training-based samplers, including knowledge distillation [34, 26] and noise level or sample trajectory learning [35, 31, 17, 41]. These methods are model-specific and thus require a possibly expensive and time-consuming training stage before they can be used for practical sampling process, which might limit their applicability and flexibility. The second category consists of training-free samplers that are suitable for all pre-trained DPMs in a simple plug-and-play manner. These training-free samplers include adopting implicit [37] or analytical [5] generation process as well as advanced differential equation solvers [38, 13, 22, 39, 24, 25]. Among these training-free methods, the commonly used method for guided sampling is DDIM [37], which is proven as a first-order diffusion ODE solver [24, 34]. Recently, dedicated high-order diffusion ODE solvers such as DPM-solver [24] and DPM-Solver++ [25] have achieved remarkable speedup performance in the field of image generation, which are adopted by Stable Diffusion [32] as default acceleration algorithms.

3 Proposed Method

After knowing the importance of accelerating diffusion models for protein structure generation, we are going to be dedicated to designing proper methods that can accelerate the diffusion process in protein structure generation. However, one key question is naturally raised: "How can we evaluate our proposed acceleration method?" As evaluation is the most fundamental component of research, and to the best of our knowledge, no prior work has explored accelerating diffusion models in protein generation, so we will first answer this question.

3.1 Evaluation Protocol

The acceleration of diffusion models are well explored by researchers in image generation, natural language generation, and other real-world applications (see section 2). In the traditional image generation and natural language generation, the basic approach to evaluate acceleration method is to track the trade-off between two targets: *sampling efficiency* and *sample quality*. The first target can be evaluated through *number of score function evaluations* (**NFE**) (i.e. number of model call in reverse diffusion process). But the sample quality is often evaluated through FID score [9] or perplexity [6] in CV or NLP, which is not available in protein structure modeling. What's more, it's easy for human to judge whether a sampled image or sentence appeal to our needs, but it's hard for human to judge by eyes whether a generated protein structure is valid according to complex physical and chemical constraints.

Based on the issue, we choose to focus on *sequence-conditioned generation*, which means we provide the information of amino acid sequence for the diffusion model, and expect the model can generate valid structures based on the given sequence. Since we are accessible to the ground truth structure conditioned on amino acid sequence, we can evaluate the *sample quality* trough comparison between the model generated structures and ground truth structures discovered and labelled in biological experiments. Here we choose *The Local Distance Difference Test* (**lDDT**) [28] as the comparison metric.

lDDT. The lDDT measure evaluates the accuracy of a protein model in reproducing the environment of a reference structure. It is calculated by comparing the distances between pairs of atoms in the reference structure that are within a specified distance (called the inclusion radius) and not belonging to the same residue. These distances make up a set called L. The lDDT score is determined by calculating the fraction of distances in L that are preserved in the model, meaning that they are within a certain tolerance of the corresponding distance in the reference structure. If one or both atoms defining a distance in L are not present in the model, the distance is considered non-preserved. The lDDT score can be calculated using all atoms in the model, or just the distances between alpha carbon atoms or backbone atoms. The lDDT score reported in this report will be based on alpha carbon atoms, i.e. C_α-lDDT.

Above all, we are able to evaluate our proposed acceleration methods based on NFE and lDDT as the protocols to *sampling efficiency* and *sample quality*. We expect a good acceleration method to have both small NFE and lDDT.

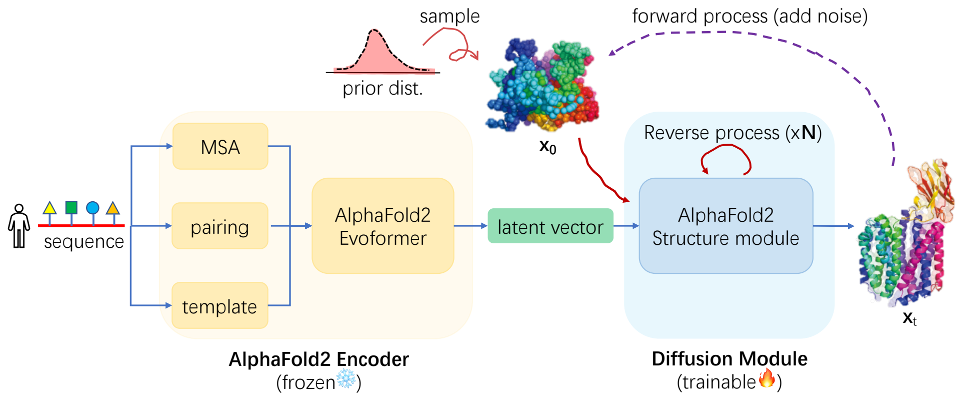

Figure 1: Our proposed diffusion model. The model is provided with a certain amino acid sequence as input, and generate *multi-sequence alignment*(MSA), *paring*, and *template* features that are used by AF2 Evoformer. We use a pre-trained AF2 evoformer to encode sequence feature into a latent vector $\boldsymbol{z}$, then feed $\boldsymbol{z}$ into the diffusion model together with noisy structures $\boldsymbol{x}_0$. The diffusion model is built upon AF2 structure module which is trainable during training.

3.2 Diffusion Model

Overview. Although there exists some diffusion models applied in protein structure generation(as shown in 2), the works on sequence-conditioned generation is rare. We design a simple and effective diffusion model based on AlphaFold2 (AF2) modules (presentd in 1). To condition on sequence information, we use pre-trained AF2 Evoformer as feature extractor and follows AF2's preprocessing techniques to prepare *multi-sequence alignment* (MSA), *paring*, and *template* features. Based on the input sequence, the pre-trained Evoformer outputs a dense latent vector $\boldsymbol{z}$ as a representation of the given sequence. Then we adopts AF2 structure module as backbone architecture for our structure diffusion model, which takes noisy structure $\boldsymbol{x}_0$ and sequence representation $\boldsymbol{z}$, and output the denoised structure $\boldsymbol{x}_t$. Since the architecture is almost the same as AF2, we don't explain a lot in this report but point out only the differences.

3.2.1 Definitions and Notations

Before going into the details of our proposed diffusion processes, we first introduces notations used throughout the report and formally states the problem. An amino acid in a protein complex can be represented by its type, C_α atom coordinate, and the orientation, denoted as $s_i \in \{\texttt{ACDEFGHIKLMNPQRSTVWY}\}, \boldsymbol{x}_i \in \mathbb{R}^3, \boldsymbol{O}_i \in SO(3)$, respectively. Here $i = 1, \cdots n$, and n is the number of amino acids in the protein complex. Formally, our goal of sequence-conditioned generation is to model the distribution of $\{\boldsymbol{x}_i, \boldsymbol{O}_i\}_{i=1}^n$ given the sequence information $\{s_i\}_{i=1}^n$, i.e. $\mathbf{P}(\{\boldsymbol{x}_i, \boldsymbol{O}_i\}_{i=1}^n | \{s_i\}_{i=1}^n)$ or $\mathbf{P}(\{\boldsymbol{x}_i, \boldsymbol{O}_i\}_{i=1}^n | \{\boldsymbol{z}_i\}_{i=1}^n)$.

3.2.2 Diffusion processes

In a typical diffusion probabilistic model, there are two Markov chains of diffusion processes. The forward diffusion process adds noise to the data until it approximately reaches the prior distribution, while the generative diffusion process starts from the prior distribution and transforms it to the desired distribution. The state of amino acid j at time step t is represented by $(\boldsymbol{x}_j^t, \boldsymbol{O}_j^t)$, with $t = 0$ representing the samples from the prior distribution and $t = N$ representing the state of the real data. The forward diffusion process goes from $t = N$ to 0, while the generative diffusion process proceeds in the opposite direction. To make it clear, we denote structure generated at time t as $\mathcal{R}^t = \{\boldsymbol{x}_i^t, \boldsymbol{O}_i^t\}_{i=1}^n$. Referred to [27], the diffusion process for the coordinates $\boldsymbol{x}_j^t$ and orientations $\boldsymbol{O}_j^t$ of amino acid j is designed as follows.

Diffusion process of C_α Coordinates. In order to make the coordinates of atoms more easily comparable, we often scale and shift the coordinates of an entire structure so that the distribution of atom coordinates approximates a standard normal distribution. We can then define the forward diffusion process for the normalized C coordinates, denoted as $\boldsymbol{x}_j$, as follows:

$$q(\boldsymbol{x}_j^{t-1}|\boldsymbol{x}_j^t) = \mathcal{N}(\boldsymbol{x}_j^{t-1}|\sqrt{1-\beta^t_{\text{pos}}} \cdot \boldsymbol{x}_j^t, \beta^t_{\text{pos}}\boldsymbol{I}), \tag{1}$$

$$q(\boldsymbol{x}_j^t|\boldsymbol{x}_j^N) = \mathcal{N}(\boldsymbol{x}_j^t|\sqrt{\bar{\alpha}^N_{\text{pos}}} \cdot \boldsymbol{x}_j^N, (1-\bar{\alpha}^N_{\text{pos}})\boldsymbol{I}), \tag{2}$$

where β^t_{pos} controls the rate of diffusion and its value increases from 0 to 1 as time step goes from 0 to t, and $\bar{\alpha}^t_{\text{pos}} = \prod_{\gamma=t+1}^{N}(1-\beta^{\gamma}_{\text{pos}})$. Using the reparameterization tricks, the generative diffusion process is defined as:

$$p(\boldsymbol{x}_j^t|\mathcal{R}^{t-1}) = \mathcal{N}(\boldsymbol{x}_j^t|\boldsymbol{\mu}_p(\mathcal{R}^{t-1}), \beta^{t-1}_{\text{pos}}\boldsymbol{I}), \tag{3}$$

$$\boldsymbol{\mu}_p(\mathcal{R}^t) = \frac{1}{\sqrt{\alpha^t_{\text{pos}}}}(\boldsymbol{x}_j^t - \frac{\beta^t_{\text{pos}}}{\sqrt{1-\bar{\alpha}^t_{\text{pos}}}} G(\mathcal{R}^t)[j]). \tag{4}$$

Here $G(\cdot)$ is parameterized using AF2 structure module as mentioned above. $G(\cdot)[j]$ predicts the standard Gaussian noise $\epsilon_j \sim \mathcal{N}(0, \boldsymbol{I})$ added to $\sqrt{\bar{\alpha}^N_{\text{pos}}}\boldsymbol{x}_j^N$ (scaled coordinate of amino acid j) based on the reparameterization of Eq. 2. The objective function of training the generative process is the expected MSE between G and ϵ_j, which is simplified from aligning distribution p to the posterior $p(\boldsymbol{x}_j^t|\boldsymbol{x}_j^{t-1}, \boldsymbol{x}_j^N)$:

$$L^t_{\text{pos}} = \mathbb{E}[\frac{1}{m}\sum_{j=1}^{n}||\epsilon_j - G(\mathcal{R}^t, \boldsymbol{z})||^2] \tag{5}$$

Diffusion process of Amino Acid Coordinates $\boldsymbol{O}_j$. In order to learn and generate amino acid orientations, which are represented by elements of the special orthogonal group SO(3), [18] propose an iterative perturbation-denoising scheme. While this scheme is based on the overall principle of diffusion probabilistic models, it does not strictly follow the framework of diffusion processes. The distribution of orientations perturbed for t steps is defined as follows:

$$q(\boldsymbol{O}_j^t|\boldsymbol{O}_j^N) = \mathcal{IG}_{SO(3)}(\boldsymbol{O}_j^t|\text{ScaleRot}(\sqrt{\bar{\alpha}^t_{ori}}, \boldsymbol{O}_j^N), 1-\bar{\alpha}^t_{\text{ori}}). \tag{6}$$

$\mathcal{IG}_{SO(3)}$ denotes the isotropic Gaussian distribution on SO(3) parameterized by a mean rotation and a scalar variance. ScaleRot modifies the rotation matrix by scaling its rotation angle with the rotation axis fixed. $\bar{\alpha}^t_{\text{ori}} = \prod_{\gamma=t+1}^{N}(1-\beta^{\gamma}_{\text{ori}})$, where β^t_{ori} is the variance increases with time step t. The conditional distribution used for the generation process of orientations is defined as:

$$p(\boldsymbol{O}_j^t|\mathcal{R}^{t-1}) = \mathcal{IG}_{SO(3)}(\boldsymbol{O}_j^t|H(\mathcal{R}^{t-1}, \boldsymbol{z})[j], \beta^{t-1}_{\text{ori}}), \tag{7}$$

In the above expression, the function $H(\cdot)[j]$ is parameterized by the structure module and is used to denoise the orientation of amino acid j, outputting the denoised orientation matrix. To train the conditional distribution, we need to align the predicted orientation from $H(\cdot)$ with the real orientation. To do this, we define the training objective as minimizing the expected discrepancy between the real and predicted orientation matrices, which is measured by the inner product between these matrices. This can be expressed as follows:

$$L^t_{\text{ori}} = \mathbb{E}[\frac{1}{m}\sum_{j=1}^{n}||(\boldsymbol{O}_j^N)^T\hat{\boldsymbol{O}}_j^t - I||_F^2], \tag{8}$$

where $\hat{\boldsymbol{O}}_j^t = H(\cdot)[j]$ is the predicted orientation for amino acid j.

Overall Training Objective. By summing the above equations, we obtain the final training objective function as:

$$L = \mathbb{E}_{t \sim U(1 \cdots T)}[L_{\text{ori}}^{t} + L_{\text{pos}}^{t}] \tag{9}$$

Note that our sequence feature $\boldsymbol{z}$ is fixed during training and inference since the feature extractor is frozen. This prevents the interference from other parts of model, which allows us focus only on our diffusion model part.

3.3 Diffusion Module

Our diffusion module is built upon AF2 structure module. Firstly we encode time step information into the latent vector $\boldsymbol{z}$ by

$$\hat{\boldsymbol{z}} = \boldsymbol{z} + \mathrm{MLP}([\boldsymbol{z}, \mathrm{Embedding(t)}]) \tag{10}$$

Here t is the discrete time step from 0 to N. We construct an time embedding layer to encode time step information in diffusion processes, and use a multi-layer perceptron to fuse time information into the latent vector. Since the vanilla AF2 structure module takes a latent vector and noisy structure (coordinates and orientations) as input, we simply substitute the latent vector as $\hat{\boldsymbol{z}}$ and the noisy structure as $\{\boldsymbol{x}_i^t, \boldsymbol{O}_i^t\}_{i=1}^n$ defined in 3.2.2.

3.4 Acceleration Method

We test the speedup performance of DDIM [37], DPM-solver [24] and DPM-Solver++ [25] on the task of protein structure generation based on the model architecture introduced in 3.2. As is stated in 2.2, DDIM is proven to be a first-order diffusion ODE solver, while both DPM-Solver and DPM-Solver++ are high-order diffusion ODE solvers. Specifically, for general DPMs, consider the *forward process* $\{\boldsymbol{x}_t\}_{t \in [0,T]}$ with $T > 0$ starting with $\boldsymbol{x}_0$, such that for any $t \in [0, T]$, the distribution of $\boldsymbol{x}_t$ conditioned on $\boldsymbol{x}_0$ satisfies

$$q_{0t}(\boldsymbol{x}_t | \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t | \alpha(t)\boldsymbol{x}_0, \sigma^2(t)\boldsymbol{I}) \tag{11}$$

where $\alpha(t)$, $\sigma(t) \in \mathbb{R}^+$ are differentiable functions of t with bounded derivatives, and we denote them as α_t, σ_t for simplicity. Kingma et al. [15] prove that for any $t \in [0, T]$, the following stochastic differential equation (SDE) has the same transition distribution $q_{0t}(\boldsymbol{x}_t | \boldsymbol{x}_0)$ as in Eq. 11

$$\mathrm{d}\boldsymbol{x}_t = f(t)\boldsymbol{x}_t \mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}_t \tag{12}$$

where $\boldsymbol{w}_t \in \mathbb{R}^D$ is the standard Wiener process, and

$$f(t) = \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}, \ \ g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2 \tag{13}$$

Song et al. [38] show that the forward process in Eq. 12 has an equivalent *reverse process* from T to 0, which is described by the *diffusion SDE*

$$\mathrm{d}\boldsymbol{x}_t = \Big[f(t)\boldsymbol{x}_t + \frac{g^2(t)}{\sigma_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)\Big]\mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}_t, \ \ \boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \tilde{\sigma}^2\boldsymbol{I}) \tag{14}$$

in which $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ parameterized by θ is a neural network that estimates the scaled score function $-\sigma_t \nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x}_t)$. Song et al. [38] prove that the following parameterized ODE, which is defined as the *diffusion ODE*, has the same marginal distribution at each time step t as that of the diffusion SDE

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}_\theta(\boldsymbol{x}_t, t) := f(t)\boldsymbol{x}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t), \ \ \boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \tilde{\sigma}^2\boldsymbol{I}) \tag{15}$$

Comparing to SDEs, diffusion ODEs can be solved with larger step sizes as they have no randomness. And it's worth noting that DDIM, DPM-Solver and DPM-Solver++ all take advantage of efficient numerical ODE solvers to accelerate the sampling process. To avoid possible redundancy, here we present the algorithm of DPM-Solver as an example.

Note that the R.H.S. of Eq.15 has a *semi-linear* structure, *i.e.* the part $f(t)\boldsymbol{x}_t$ is a linear function of $\boldsymbol{x}_t$. By *variation of constants* formula, we have

$$\boldsymbol{x}_t = e^{\int_t^s f(\tau)d\tau}\boldsymbol{x}_s + \int_s^t \Big(e^{\int_t^s f(\tau)d\tau}\frac{g^2(\tau)}{2\sigma_\tau}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau)\Big)\mathrm{d}\tau \tag{16}$$

We introduce $\lambda_t := \log(\alpha_t/\sigma_t)$ and rewrite 16 by *change-of-variable* for λ, then we have

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \alpha_t \int_{\lambda_s}^{\lambda_t} e^{-\lambda}\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)\mathrm{d}\lambda \tag{17}$$

We apply $(k-1)\text{-th}$ order Taylor Expansion to the neural network term and compute the integral *analytically* using integration-by-parts to obtain **DPM-Solver-k** for a specific order k.

$$\begin{aligned}\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i}\sum_{n=0}^{k-1}\hat{\boldsymbol{\epsilon}}_\theta^{(n)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}})\int_{\lambda_{t_{i-1}}}^{\lambda_{t_i}} e^{-\lambda}\frac{(\lambda-\lambda_{t_{i-1}})^n}{n!}\mathrm{d}\lambda \\ + \mathcal{O}((\lambda_{t_i}-\lambda_{t_{i-1}})^{k+1})\end{aligned} \tag{18}$$

In practice, we take $k = 2$ and use DPM-Solver-2 as one of our acceleration method. The algorithm of DPM-Solver++ is highly similar to the original DPM-Solver, except for solving diffusion ODEs of $\boldsymbol{x}_\theta$ rather than $\boldsymbol{\epsilon}_\theta$. We test DPM-Solver-2S (second-order single-step) and DPM-Solver-2M (second-order multi-step) on the protein structure generation task.

4 Experiment

Training Details. Our model is implemented in PyTorch and trained on 8 GPU Nvidia A100 80G for 3000 steps (roughly 3 hours). We use AdamW [23] algorithm as our optimizer. The learning rate is set to 5e-4 and all the other parameters of AdamW stay as default in PyTorch. The batch size is set to 1 according to the memory limitation. In addition, gradients of all parameters are clipped based on the norm at 0.3 to address the spikes of the loss and gradient norms during training. Default precision format in Nvidia A100 GPUs is set to TensorFloat-32 for matrix operations. The dimension for time embedding is set to 128. The total time step N is set to 1000. Other parameters of the model keep the same as vanilla AF2.

vanilla	1000 NFE	500 NFE	100 NFE	50 NFE	25NFE	1 NFE
LDDT(average)	0.824	0.820	0.818	0.8	0.74	0.30
LDDT(7BI4_A)	0.70	0.685	0.676	0.65	0.45	0.18
time(CAMEO)/s	42331	21888	5356	3279	2321	1273
time(7BI4_A)/s	1257.3	648.2	158.7	96.9	69.8	40.1

Table 1: Relusts of the vanilla diffusion model without ant accelerated method.

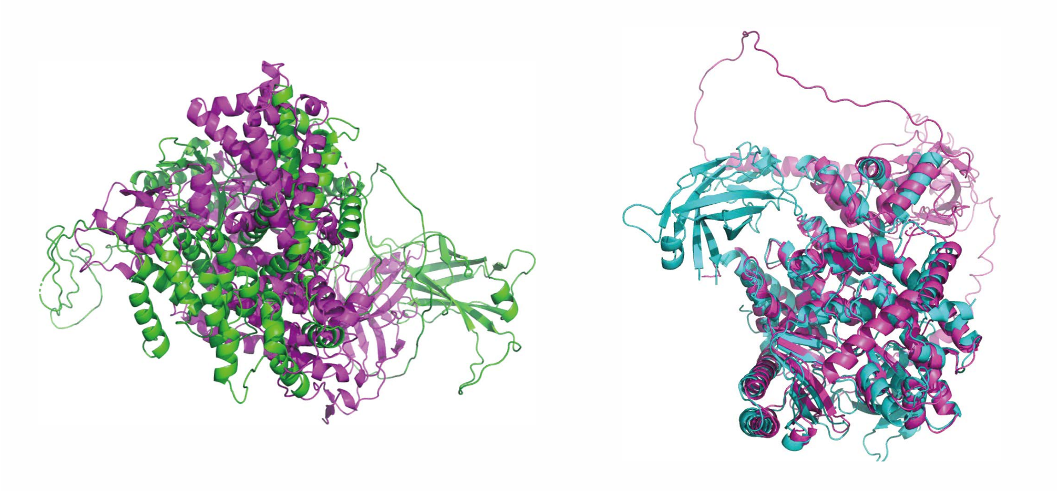

Figure 2: The 3D structure of real and predicted 7BI4_A. The pink image show the real 3D structure. The green image on the left is the result of the vanilla diffusion model at 50 NFE with $lddt = 0.65$ and $time = 96.9s$. The cyan image on the right is the results at 1000 NFE with $lddt = 0.70$ and $time = 1267.3s$.

Test Data and Results. We choose CAMEO which consists 146 of the most recent single-chain proteins as our test data set. We record the lddt at various NFEs. The results of the vanilla diffusion model as the baseline performance are shown in Table 1. We find that our baseline converges at about 1000 NFE. To save time, we select 7BI4_A, the PDB id of the protein, to run DDIM, DPM-Solver(2nd), DPM-Solver++(2S) and DPM-Solver(2M). Its real 3D structure and predicted structure of the baseline are shown in Figure 3. It's clear that our model rebuild 7BI4_A successfully and when LDDT reaches 0.70, the relust is good enough. As shown in Figure 3, DPM-Solver++(2S/2M) converges at almost 40 NFE, almost 60s. DPM-Solver++(2S/2M) achieve the best speedup performance. DPM-Solver(2nd) performs slightly worse than DPM-Solver++ but beats DDIM. DDIM's LDDT doesn't achieve 0.7 before 320 NFE and it keep increasing with the NFE. So we can't prove DDIM perform better than our baseline.

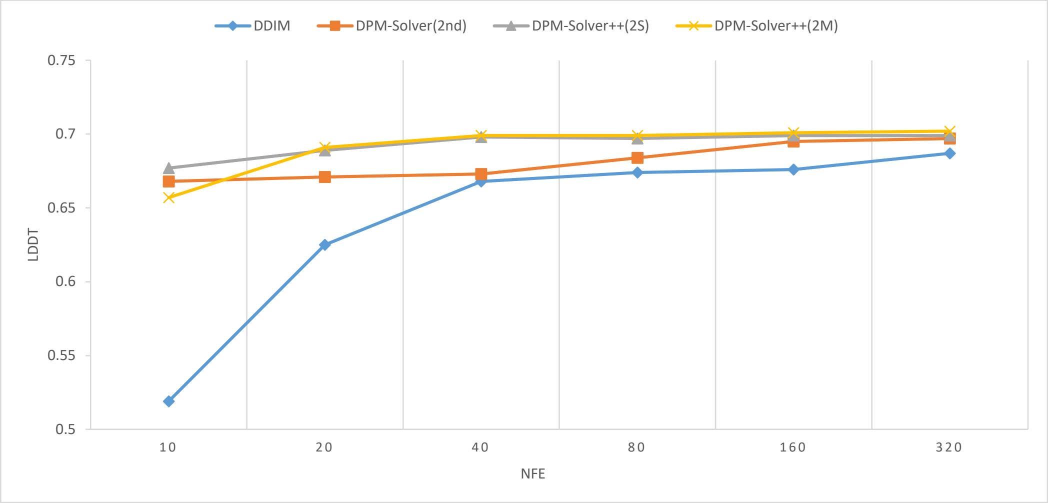

Figure 3: Comparison of various acceleration methods. This image shows how the lDDT of different acceleration methods change with NFE increasing.

5 Conclusion

We study accelerated methods on our diffusion-based sequence-conditioned generative model. We prove that our model can achieve the same lddt as the vanilla diffusion model. Besides, we demonstate that the acceleration methods significantly improve the convergence speed of our model by a large margin. It could reduce the time to almost one twentieth of the original. These methods work by speeduping the sampling process. Experiments on 7BI4_A show that DPM-Solver++ converges at most 40 NFE.

This will encourage us to explore more accelerated methods in protein design. However, we need to be careful in this course because not all the accelerated general methods will perform well in this field, taking DDIM as an example.

Author Contribution

All authors conceived and designed the work. H.L. implemented the main part of the model. S.L. and Y.W. performed research on accelerated sampling algorithms. All authors analysed the result and wrote the manuscript.

References

[1] R. F. Alford, A. Leaver-Fay, J. R. Jeliazkov, M. J. O'Meara, F. P. DiMaio, H. Park, M. V. Shapovalov, P. D. Renfrew, V. K. Mulligan, K. Kappel, et al. The rosetta all-atom energy function for macromolecular modeling and design. *Journal of chemical theory and computation*, 13(6):3031–3048, 2017.

[2] N. Anand, R. Eguchi, and P.-S. Huang. Fully differentiable full-atom protein backbone generation. 2019.

[3] N. Anand, R. Eguchi, I. I. Mathews, C. P. Perez, A. Derry, R. B. Altman, and P.-S. Huang. Protein sequence design with a learned potential. *Nature communications*, 13(1):1–11, 2022.

[4] M. Baek, F. DiMaio, I. Anishchenko, J. Dauparas, S. Ovchinnikov, G. R. Lee, J. Wang, Q. Cong, L. N. Kinch, R. D. Schaeffer, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021.

[5] F. Bao, C. Li, J. Zhu, and B. Zhang. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. *arXiv preprint arXiv:2201.06503*, 2022.

[6] S. F. Chen, D. Beeferman, and R. Rosenfeld. Evaluation metrics for language models. 1998.

[7] R. Das and D. Baker. Macromolecular modeling with rosetta. *Annu. Rev. Biochem.*, 77:363–382, 2008.

[8] R. R. Eguchi, C. A. Choe, and P.-S. Huang. Ig-vae: Generative modeling of protein structure by direct 3d coordinate generation. *Biorxiv*, pages 2020–08, 2022.

[9] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[10] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[11] P.-S. Huang, S. E. Boyken, and D. Baker. The coming of age of de novo protein design. *Nature*, 537(7620):320–327, 2016.

[12] J. Ingraham, M. Baranov, Z. Costello, V. Frappier, A. Ismail, S. Tie, W. Wang, V. Xue, F. Obermeyer, A. Beam, et al. Illuminating protein space with a programmable generative model. *bioRxiv*, 2022.

[13] A. Jolicoeur-Martineau, K. Li, R. Piché-Taillefer, T. Kachman, and I. Mitliagkas. Gotta go fast when generating data with score-based models. *arXiv preprint arXiv:2105.14080*, 2021.

[14] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[15] D. Kingma, T. Salimans, B. Poole, and J. Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

[16] B. Kuhlman and D. Baker. Native protein sequences are close to optimal for their structures. *Proceedings of the National Academy of Sciences*, 97(19):10383–10388, 2000.

[17] M. W. Lam, J. Wang, R. Huang, D. Su, and D. Yu. Bilateral denoising diffusion models. *arXiv preprint arXiv:2108.11514*, 2021.

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

[19] A. Leaver-Fay, M. J. O'Meara, M. Tyka, R. Jacak, Y. Song, E. H. Kellogg, J. Thompson, I. W. Davis, R. A. Pache, S. Lyskov, et al. Scientific benchmarks for guiding macromolecular energy function improvement. In *Methods in enzymology*, volume 523, pages 109–143. Elsevier, 2013.

[20] A. Leaver-Fay, M. Tyka, S. M. Lewis, O. F. Lange, J. Thompson, R. Jacak, K. W. Kaufman, P. D. Renfrew, C. A. Smith, W. Sheffler, et al. Rosetta3: an object-oriented software suite for the simulation and design of macromolecules. In *Methods in enzymology*, volume 487, pages 545–574. Elsevier, 2011.

[21] J. S. Lee and P. M. Kim. Proteinsgm: Score-based generative modeling for de novo protein design. *bioRxiv*, 2022.

[22] L. Liu, Y. Ren, Z. Lin, and Z. Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

[23] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[24] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *arXiv preprint arXiv:2206.00927*, 2022.

[25] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

[26] E. Luhman and T. Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

[27] S. Luo, Y. Su, X. Peng, S. Wang, J. Peng, and J. Ma. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[28] V. Mariani, M. Biasini, A. Barbato, and T. Schwede. lddt: a local superposition-free score for comparing protein structures and models using distance difference tests. *Bioinformatics*, 29(21):2722–2728, 2013.

[29] R. Mariuzza, S. Phillips, and R. Poljak. The structural basis of antigen-antibody recognition. *Annual review of biophysics and biophysical chemistry*, 16(1):139–159, 1987.

[30] A. Nichol, H. Jun, P. Dhariwal, P. Mishkin, and M. Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022.

[31] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[32] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[33] S. Sabban and M. Markovsky. Ramanet: Computational de novo helical protein backbone design using a long short-term memory generative neural network. *bioRxiv*, page 671552, 2020.

[34] T. Salimans and J. Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[35] R. San-Roman, E. Nachmani, and L. Wolf. Noise estimation for generative diffusion models. *arXiv preprint arXiv:2104.02600*, 2021.

[36] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[37] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[38] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[39] H. Tachibana, M. Go, M. Inahara, Y. Katayama, and Y. Watanabe. It\^{o}-taylor sampling scheme for denoising diffusion probabilistic models using ideal derivatives. *arXiv preprint arXiv:2112.13339*, 2021.

[40] B. L. Trippe, J. Yim, D. Tischer, T. Broderick, D. Baker, R. Barzilay, and T. Jaakkola. Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem. *arXiv preprint arXiv:2206.04119*, 2022.

[41] D. Watson, W. Chan, J. Ho, and M. Norouzi. Learning fast samplers for diffusion models by differentiating through sample quality. In *International Conference on Learning Representations*, 2021.

[42] J. L. Watson, D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, et al. Broadly applicable and accurate protein design by integrating structure prediction networks and diffusion generative models. *bioRxiv*, 2022.

[43] K. E. Wu, K. K. Yang, R. v. d. Berg, J. Y. Zou, A. X. Lu, and A. P. Amini. Protein structure generation via folding diffusion. *arXiv preprint arXiv:2209.15611*, 2022.

[44] J. Yang, I. Anishchenko, H. Park, Z. Peng, S. Ovchinnikov, and D. Baker. Improved protein structure prediction using predicted interresidue orientations. *Proceedings of the National Academy of Sciences*, 117(3):1496–1503, 2020.

[45] Q. Zhou, P. Zhou, A. L. Wang, D. Wu, M. Zhao, T. C. Südhof, and A. T. Brunger. The primed snare–complexin–synaptotagmin complex for neuronal exocytosis. *Nature*, 548(7668):420–425, 2017.